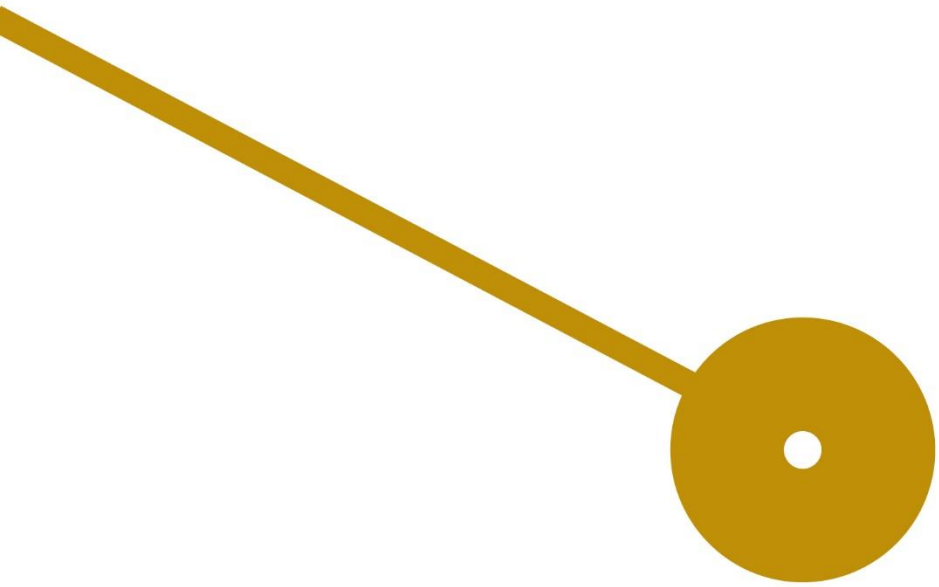
ESCOLA
SUPERIOR
DE MÚSICA
E ARTES
DO ESPETÁCULO
POLITÉCNICO
DO PORTO

MESTRADO
TEATRO
Design de Luz

Sistema de rastreamento interativo para iluminação cénica como ferramenta de criação

Élio Silva Moreira

9/2017



ESMAE
ESCOLA SUPERIOR DE MÚSICA E ARTES DO ESPETÁCULO
POLITÉCNICO DO PORTO

P.PORTO

M MESTRADO
TEATRO
Design de Luz

Sistema de rastreamento interativo para iluminação cénica como ferramenta de criação

Élio Silva Moreira

Projeto apresentado à Escola Superior de Música e Artes do Espetáculo como requisito parcial para obtenção do grau de Mestre em Teatro, especialização Design de Luz.

Professor Orientador
Pedro Moreira Cabral

9/**2017**

Agradecimentos

À minha família pelo apoio e incentivo constante.

Ao meu orientador Pedro Cabral por todo o apoio, desafios lançados e amizade.

À Catarina Ferreira pela participação e sobretudo paciência.

À Alexandra Prezado pela colaboração e constante motivação.

A todos os meus docentes pela cooperação e compreensão.

À equipa técnica do Teatro Sá de Miranda pelo encorajamento constante e auxílio.

Aos meus colegas de curso pelo companheirismo e incentivos.

Resumo

Este projeto visa a criação de uma interface, de baixo custo, capaz de controlar a iluminação cénica, em tempo-real, em função da posição, orientação e velocidade de rotação de um performer, objetivando a interatividade entre estes.

Assim, procedeu-se à enumeração e comparação dos principais sistemas de rastreamento performativo de modo a compreender quais as vantagens de cada um e quais as características interativas existentes.

Posto isso, seguiu-se a planificação de todo o desenvolvimento da interface, desde a escolha de ferramentas à aplicação da interface. Já no seu desenvolvimento, é apresentado e explicado o percurso tomado e quais as dificuldades encontradas. São igualmente expostas as funcionalidades de rastreamento e interação que a interface criada fornece.

Por fim, foi analisada toda a fase de criação e a aplicabilidade do sistema de forma a compreender de que forma este tipo de interfaces munem o profissional de luz e quais as novas possibilidades na criação artística.

Palavras-chave

Interatividade; interface; iluminação cénica; performance; captura de movimento em tempo-real, tracking.

Abtract

This project aims the development of a low-cost interface that reacts in real-time to a performer's position, orientation and rotation speed thru autonomous stage lighting control in order to get interactive comunication.

Therefore, main stage tracking systems were identified and compared each other to aquire their advantages and existing interactive features.

Then, development's planning was carried out from tools selection to the interface's application. All development's choices are then presented and explained as well as dificulties. Tracking and interactivity features provided from the created interface were here clarified too.

Lastly, there is an analysis from interface's creation to presentation in order to understand how can these interfaces equip lighting practicioners and what does it introduce to existing artistic creations.

Keywords

Interactivity; interface; stage lighting; performance; real-time motion capture; tracking.

Índice

Lista de Figuras

Lista de Ilustrações

Lista de Tabelas

Lista de Diagramas

Lista de Gráficos

1. Introdução

A interatividade é definida quando existe um elemento que reage ao estímulo de um outro. Assim, pode dizer-se que toda a arte é interativa, tendo em vista que as representações artísticas surgem de um conjunto de ações-reações que o criador artístico mantém com a obra, o intérprete e o seu público.

Hoje em dia, podem verificar-se bastantes formas de interação entre a luz e determinado performer. No entanto, essa interação pode exigir um investimento que limita à partida o objeto artístico. A necessidade de criação *in loco*, a impossibilidade de virtualização de certas interações e a inexistência de uma interface bem desenvolvida e consistente de criação de interação de luz, são muitas vezes os principais limitadores da criação artística.

De que forma é possível criar um objeto artístico com relações interativas entre a luz e performer? Que ferramentas existem e de que modo permitem alcançar essa interatividade? E como é que esses métodos podem afetar a relação que o profissional de iluminação mantém com o performer?

Estas são algumas das questões de partida para o desenvolvimento deste projeto.

Na criação artística, a criação de relações interativas com a luz é limitada, em muitos casos, por fatores técnicos. Os métodos e técnicas de programação convencionais não permitem uma resposta instantânea a impulsos dados pelo ator, bailarino ou outro. Isto acontece fundamentalmente por não existir uma evolução significativa na forma como se programa uma mesa de luz, tendo maioritariamente estagnado no modelo convencional de memórias e sequências de memórias de luz, como explica Hunt(2013): "This model privileges the static over the dynamic, and the predesigned over the immediacy of the moment in performance" (p. 3). Apesar de existirem técnicas que criam uma ilusão de interação, estas podem tornar-se bastante complexas, originando técnicas de operação que exigem um grande investimento na programação de cada interação.

Assim, assumiu-se a necessidade de recorrer à tecnologia para preencher essas lacunas. Com isso, surge o conceito de *tracking*, que permite a perseguição automática de indivíduos em palco. Infelizmente os sistemas de *tracking* atuais, além de serem bastante caros, possuem um leque de funcionalidades de interação bastante restrito, o que levou à procura de uma solução hibrida que permitisse o rastreamento de um performer para a perseguição e interação de luz.

Com isto, os principais objetivos deste projeto são:

- Desenvolver um sistema-protótipo de rastreamento de baixo custo, dotado de funcionalidades de perseguição e interação de luz;
- Analisar os resultados da aplicabilidade desse sistema no processo criativo de uma performance real;

Desta forma, iniciou-se o desenvolvimento de uma interface capaz de munir o profissional de iluminação com capacidades, programáveis, de controlo de iluminação autónoma. O principal estímulo de interação que se pretendeu implementar neste projeto foi a localização do performer. Assim, a luz deveria reagir à sua posição de forma programada, criando uma interação em tempo-real, mas acabou por se possibilitar a utilização acrescida da velocidade de rotação e a orientação de um performer como inputs de reação.

2. Estado da Arte

2.1 Evolução histórica

Antes de entrar no desenvolvimento do projeto em si, foi necessária uma investigação para compreender o que já existe no mercado das artes do espetáculo e determinar o que é que este projeto pode realmente trazer de novo. Como tal, verificou-se que um dos primeiros equipamentos de rastreamento cénico surgiu em 1975, um sistema desenvolvido pelo Institute on Technical Cybernetics, usado no Teatro Trøndelag da cidade de Trondheim, Noruega. Este sistema era dotado de um emissor de ultrassons e diversos recetores que comunicavam com um dispositivo que processava as distâncias de cada instrumento e, através de fórmulas matemáticas convertia os dados recebidos em valores DMX para os atributos *pan* e *tilt* de cada robô. Sendo um dos primeiros sistemas a ser desenvolvido, este era provido de inúmeros problemas como refere Kent (2001): "The instruments worked until someone in the audience opened a bag of chips, causing interference that affected the instruments' zoom and *tracking* abilities." (p. 15). Depois desse, sistemas mais viáveis surgiram, começando pelo Autopilot da Wybron, passando pelo Martin Light Director, pelo Zactrack, o Arttrack, o Coolux e o Blacktrax, sendo este último o mais conhecido e usado da atualidade provavelmente por ter sido desenvolvido pela mesma entidade criadora do célebre WYSIWYG, a CAST. Todos estes sistemas assentam na mesma base de configuração: pelo menos um emissor por cada indivíduo e vários recetores espalhados pelo palco. As tecnologias utilizadas vão desde ultrassons, infravermelhos, e o uso de sensores de aceleração e, todas têm as suas vantagens que levam à escolha final pela comunidade. Especialmente nos últimos dois anos emergiu uma tecnologia que usa a banda ultra-larga (*UltraWide Band* - UWB) para comunicações e rastreamento de precisão. Esta é uma tecnologia promissora, mas que ainda não está bem definida no mercado em geral. No entanto, dos sistemas de *tracking* performativo investigados, o Zactrack é o único que utiliza esta tecnologia.

Até aos dias de hoje foram desenvolvidos projetos que trabalham a luz de forma pouco convencional como o "Convergence" da Backstage Academy, que utiliza a perseguição das mãos e cabeça da performer, ou o "Pattern Recognition" de Memo Akten que utiliza o rastreamento de padrões de movimento do corpo em tempo real como origem do movimento da robótica de iluminação.

2.2 Métodos de localização

2.2.1 Técnicas de localização

Através da matemática, álgebra, física e outros campos científicos é possível criar métodos que são utilizados por controladores eletrónicos no cálculo da localização de dispositivos emissores de ondas de radiofrequência. A trilateração e a triangulação são os dois métodos mais utilizados.

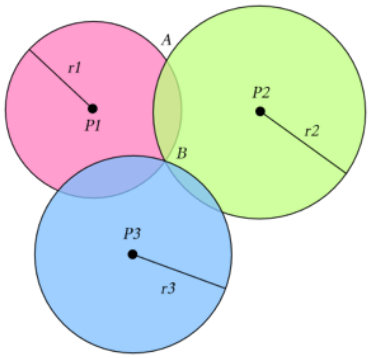


Figura 1- Exemplo de uso do método da trilateração

Técnicas como o *Received Signal Strength* (RSS) ou o *Time Deference of Arrival* (TDoA) são usadas como recurso primário para o método matemático de localização através da medição de distancias, a trilateração. O RSS usa a proporcionalidade não linear da energia das ondas eletromagnéticas em função da distância para calcular a distância entre dois dispositivos, o TDoA utiliza a diferença de tempo que uma onda eletromagnética levou para percorrer o caminho entre dois dispositivos para calcular essa distância.

O *Angle of Arrival* (AoA) utiliza antenas direcionais para detetar a proveniência do sinal eletromagnético mais forte, identificando assim a direção do dispositivo emissor necessária para o método matemático de triangulação que se baseia nos ângulos detetados por vários dispositivos.

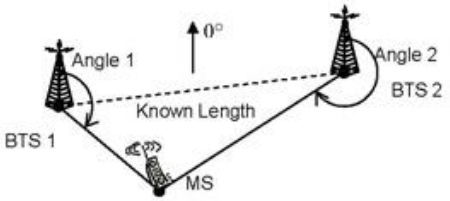


Figura 2- Exemplo do uso do método de triangulação

2.2.2 Tecnologias de localização

Antes de conhecer os vários sistemas existentes na indústria da iluminação é importante fazer uma breve introdução às diferentes tecnologias e técnicas de rastreamento mais usadas no mercado em geral. Todas elas têm em comum o aproveitamento ou mesmo a modulação[1] de ondas eletromagnéticas, como por exemplo ondas de rádio, infravermelhos ou mesmo a luz visível.

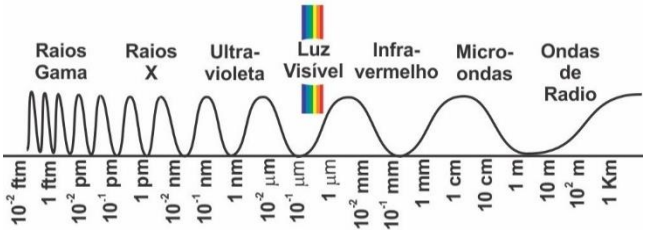


Figura 3 - Espectro de ondas eletromagnéticas

[1] A modulação de ondas eletromagnéticas é o processo de deformação de características como a amplitude, intensidade, frequência ou o comprimento de onda, de forma a enviar informação para um recetor que a irá demodular.

GPS - Global Positioning System

Este sistema foi criado somente com objetivos militares, mas posteriormente disponibilizado para uso civil. O sistema faz uso de 24 satélites em órbita da Terra que são usados para calcular a posição bidimensional e altura através de um mínimo de quatro satélites. Com esta técnica, e com o uso incremental de satélites disponíveis no momento é possível calcular a posição com um erro relativo de cerca de 9 metros ou de 1 metro, se adicionadas referências terrestres aos cálculos.

IMU - Inertial Measurement Unit

É uma unidade que aloca sensores como acelerómetros, giroscópios e até magnetómetros. Cada um destes sensores possui uma quantidade de ângulos de liberdade, por exemplo, o acelerómetro pode possuir três ângulos de liberdade: aceleração em x, y e z. Assim, um IMU pode ter 6, 9, ou até mesmo 12 ângulos de liberdade, dependendo dos sensores embutidos.

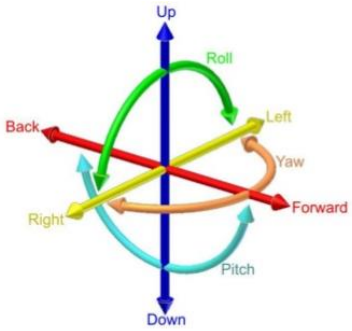


Figura 4 - Principais ângulos de liberdade de um IMU

Com este dispositivo é possível retirar-se com grande precisão e rapidez dados de orientação, aceleração, velocidade angular e posicionamento do corpo associado ao IMU. A grande desvantagem é que os erros de medição são acumulados, ou seja, vão somando ao longo do tempo, resultando um desvio exponencial da medição relativamente à posição real.

Imagem térmica

Através de uma câmara sensível a ondas infravermelhas e, através do processamento constante do vídeo, é possível rastrear alvos através do calor que emitem. Esta é uma técnica bastante simples de aplicar, mas que vem com custos acrescidos de material e com desvantagens como a débil distinção de alvos quando próximos uns dos outros ou mesmo a necessidade dos alvos estarem sempre em linha de vista das câmaras.

Figura 5 - Exemplo de imagem térmica

> *"…we still don't know if that infrared signature is 100% unique to that individual. Complicating things further, the clothing we wear, the food we eat, and our current level of activity all have a bearing on our ability to discriminate one person from the next."* (Frey, 2013)

CV – Computer Vision

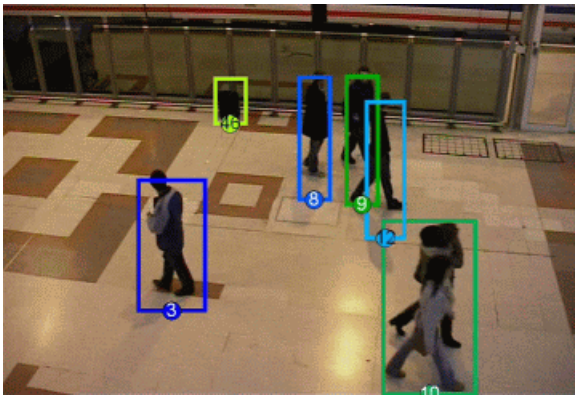


Figura 6 - Deteção de pessoas por CV

Computer Vision é um campo multidisciplinar no qual se refere à análise de imagens ou vídeo para proceder à extração e interpretação de informação objetiva. O objetivo principal desta tecnologia é a automação de tarefas visuais que o ser humano possui. É muitas vezes associada a Inteligência Artificial pois, através de complexos algoritmos, e apesar da existência de deformações nas imagens ou vídeos, é capaz de reconhecer padrões e alvos com exatidão. Nos últimos anos, a esta capacidade de reconhecimento, tem-se aliado algoritmos que munem esse sistema da capacidade de aprendizagem com erros anteriores, como o OpenAI.

Como desvantagens, uma vez mais, a necessidade de linha de vista do alvo a ser rastreado, a necessidade de introduzir parâmetros de pesquisa e, os elevados custos no desenvolvimento de software capaz de detetar e identificar pessoas de modo instantâneo e sem erros.

Bluetooth

A tecnologia de rastreamento através de Bluetooth utiliza os chamados *beacons* como âncoras, que são colocados em locais afastados uns dos outros para posteriormente calcular a posição do *beacon*-alvo através da técnica de trilateração. Assim, é sempre necessário colocar as âncoras em pontos de referência conhecidos para uma localização precisa.

Atualmente a precisão desta tecnologia pode ser de poucos centímetros, mas se adicionarmos fatores como a distância, humidade, se o corpo está em movimento ou não, e o facto do corpo Humano interferir com as ondas de rádio, então a localização torna-se inviável como refere Ramsey e Robert (2014): "…within a metre of the transmitter a positioning uncertainty of only a few centimetres would be possible, however, at 10 m the ranging error would be around 5 m" (p. 202).

WPS – Wi-Fi Positioning System

Através da rede wireless que geralmente se utiliza para se ligar à internet de qualquer rede privada ou pública no mundo, é também possível rastrear a localização de um dispositivo com esta tecnologia. Apesar de não ter sido desenhado para este fim, o Wi-Fi pode ser utilizado com as mesmas técnicas que são usadas no GPS ou Bluetooth por se tratar de tecnologia de radiofrequência semelhante.

A grande desvantagem é que existe uma inconstante e relativamente baixa precisão que é agravada com o erro de medições na existência de obstáculos como paredes, objetos, ou mesmo o corpo Humano na linha de vista entre dispositivos Wi-Fi.

Ultrassons

Através da emissão de sons de muito baixas frequências e, analisando o *Time of Flight* (ToF), que é o tempo necessário para que a onda emitida volte à origem, conseguimos criar uma espécie de sonar através do ar. Este tipo de tecnologia é bastante barato e a precisão de localização pode mesmo ser inferior a 1cm numa distancia de 2 a 6 metros, em condições de laboratório. O grande problema é sem dúvida o *Multipath Effect*. Este efeito surge quando os sinais refletidos por outras superfícies, chegam praticamente ao mesmo tempo que o sinal direto, causando um somatório de sinais que resulta no cálculo erróneo da posição real. Este efeito surge em muitas tecnologias de ondas eletromagnéticas. Este tipo de tecnologia é também muito suscetível às condições de temperatura, pressão, humidade e especialmente materiais metálicos.

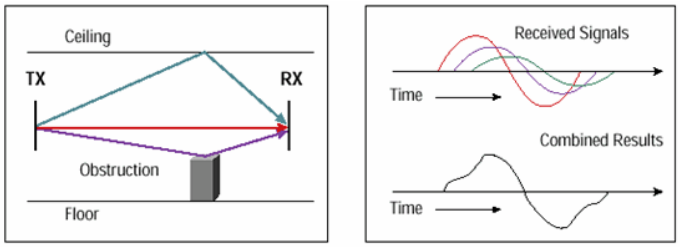


Ilustração 1 - Efeito Multipath

Infravermelhos

Esta é a tecnologia usada nos sistemas conhecidos de rastreamento na área das artes performativas como o Blacktrax ou o Coolux. Através de filtros infravermelhos em câmaras especializadas para esse efeito, é possível localizar uma *tag* emissora de ondas infravermelhas moduladas num delimitado espetro e, através de algoritmos computacionais que cruzam dados de múltiplas câmaras, consegue-se determinar a posição de diversas *tags* à velocidade que as câmaras e unidade de processamento permitirem. A complexidade do sistema, desde câmaras, emissores e algoritmos usados para computacionar a localização das *tags* são os principais fatores que levam a que este tipo de sistemas sejam muito caros, sendo maioritariamente utilizados em filmes de animação, ou mesmo em laboratórios de biomecânica e clinicas especializadas como faz o Vicon.

RFID – Radio Frequency Identification

A tecnologia RFID é amplamente usada na identificação e localização de produtos e mercadorias. Ao contrário do Bluetooth que consegue enviar dados e transmissões de voz ao mesmo tempo, com fiabilidade, e com criptografia entre dois dispositivos com esta tecnologia, o RFID apenas consegue transmitir uma quantidade limitada de informação,

geralmente relativa ao produto a que está associado, que é armazenada numa *tag*, que é obtida a partir de um leitor compatível associado a um software específico. Existem vários tipos de RFID com alcances bastante diferenciados, mas com bastante imprecisão e bastante suscetível a interferências.

RFID UWB – Radio-Frequency Identification by Ultra-wide Band

O UWB é baseado em dispositivos que emitem ondas eletromagnéticas de banda ultra-larga, ou seja, operam no espetro dos 3.1 a 10.6 giga hertz[2] (GHz). A taxa de transmissão de dados é de cerca de 500Mb por segundo e, devido à banda em que opera e do ciclo de vida da onda, esta tecnologia é praticamente imune ao *multipath effect*. Um outro fator favorável do uso desta tecnologia é o facto de que, ao operar na banda ultra-larga, a sensibilidade a interferências com outro tipo de tecnologias existentes é extremamente reduzida. Dispositivos com esta tecnologia podem ainda comunicar através de várias paredes sem qualquer perda de informação, apesar de que quantos mais obstáculos existirem, pior será a precisão de localização.

2.2.3 Diferenciação de tecnologias

Com base na pesquisa inicial foi elaborada uma tabela de comparação que ajudou na compreensão das diversas tecnologias e na escolha de uma delas para o desenvolvimento do projeto.

Tecnologia	Linha de Vista	Sens. Interfer.	Precisão	Alcance	Consumo	Custo
IMU	Não	4	4	0	5	5
Imagem térmica	Sim	3	3	4	1	1
CV	Sim	4	5	5	0	1
Bluetooth	Não	3	2	3	4	4
WPS	Não	2	2	3	3	3
Ultrassons	Sim	1	4	2	4	3
Infravermelhos	Sim	3	5	4	1	1
RFID	Não	1	2	2	5	4
UWB	Não	4	4	4	3	3

Tabela 1 - Comparativo de Tecnologias (Quanto maior o seu valor, melhor)

[2] Hertz – unidade de medida para frequência, expressa em ciclos por segundo.

É um facto que em termos individuais, uma tecnologia pode ter determinadas características, mas se adicionarmos à equação um sistema constituído pelo somatório de dispositivos semelhantes, essa tecnologia provavelmente ganha bastante mais robustez, levando a que os dados apresentados na tabela não retratem a verdade. Esta é uma comparação pessoal e relativa, que serve apenas como apontamento particular e não deve ser usada como referência padrão para outros fins.

2.3 Sistemas de rastreamento

Hoje em dia existem vários sistemas de rastreamento bastante precisos e fiáveis pensados para performances como o Blacktrax, Coolux, Zactrack e outros mais direcionados para o cinema, animação digital, ciências biomecânicas e robótica como o Vicon e o Optitrack. Para melhor compreender o funcionamento deste tipo de sistemas será detalhado o funcionamento do sistema mais conhecido no mercado das artes performativas, o Blacktrax e, um dos sistemas rivais, o Zactrack, cujo funcionamento é semelhante ao protótipo desenvolvido neste projeto.

2.3.1 Blacktrax

O Blacktrax é um dos principais sistemas usados no mundo das artes performativas pela sua fiabilidade e precisão. Das tecnologias acima descritas, este sistema utiliza duas: infravermelhos, apenas para posicionamento, e IMU para orientação, aceleração e backup de posicionamento.

2.3.1.1 Hardware

No seu sistema podemos encontrar 8 equipamentos que comunicam entre si:

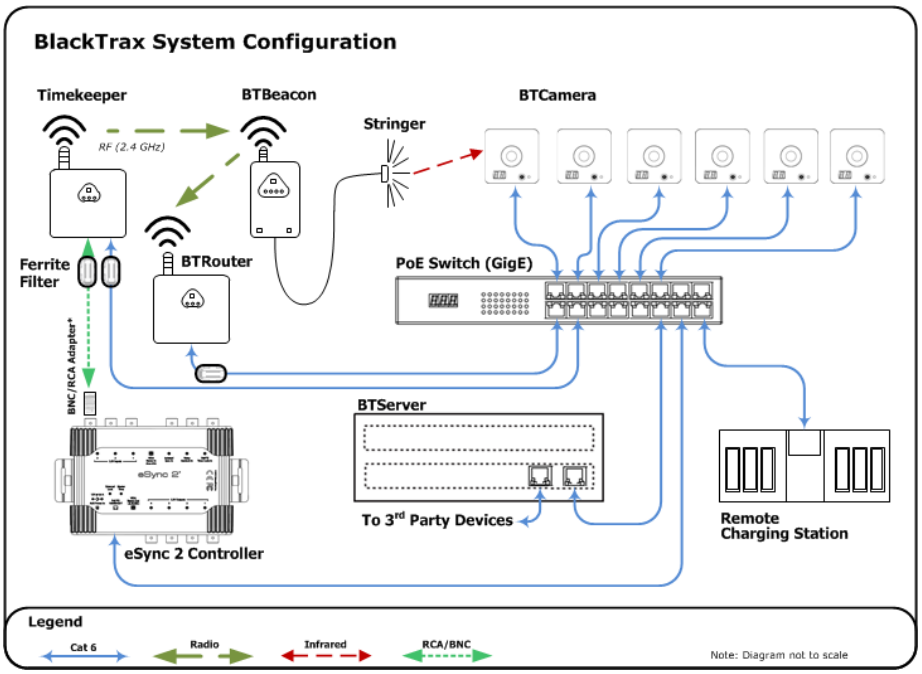


Diagrama 1 - funcionamento e comunicação entre dispositivos Blacktrax

BTBeacon – É o dispositivo que o utilizador leva consigo. Cada BTBeacon possibilita a ligação de três *Stringers*[3]. O led desses *stringers* emite pulsos de luz infravermelha com um padrão único que é detetado pelas câmaras BTCameras*, sendo a sua* posição calculada posteriormente pelo servidor BTServer.

Figura 7 - BTBeacon com Stringer acoplado

BTCamera – São câmaras dotadas com um filtro de luz infravermelha que detetam os pulsos emitidos pelos *Stringers*. Captam 100 quadros por segundo, e possuem 4 milissegundos de latência e resoluções até *FullHD*. Quanto maior a sua resolução, mais preciso é o sistema.

Figura 8 - BTCamera

Para intercomunicação de dispositivos Blacktrax é utilizado um ou mais *switches* de rede com velocidades de 1Gbps e funcionalidade *Power over Ethernet* (*PoE*).

O **BTRouter** é um router utilizado para obter informação do IMU e estado da bateria. O **TimeKeeper** e o ***eSync2***, servem essencialmente para sincronizar as BTCameras *com os* led's.

O **BTServer** é um computador de alto rendimento e é utilizado como unidade central de processamento do sistema.

[3] led infravermelho com cabo para ligação ao BTBeacon

2.3.1.2 Calibração e funcionamento

Dada a complexidade do sistema, é recomendada uma velocidade de comunicação assegurada de 1Gbps e a cabelagem de rede seja certificada com a categoria 6 (CAT6).

A montagem e orientação das câmaras é dependente dos objetos presentes no local de tal forma que as câmaras não devem perder de vista os led's dos BTBeacons. Deve ainda existir sobreposição do ângulo de visão entre câmaras para uma deteção otimizada. Após a colocação das câmaras deve-se mascarar a imagem obtida de forma a eliminar fontes de infravermelhos não desejáveis.

Após a montagem e *masking* das câmaras, procede-se à calibração do espaço tridimensional através do uso de um kit de calibração. Esse kit, ao ser movimentado pela área de rastreamento, é detetado e processado pelo BTServer que permite a calibração automática das BTCameras, determinando a distancia entre câmaras e a sua posição no espaço. Posteriormente procede-se à calibração da robótica de luz que se pretende utilizar. Essa calibração é feita individualmente de modo que quanto mais autómatos de luz existirem, mais tempo demora a calibrar.

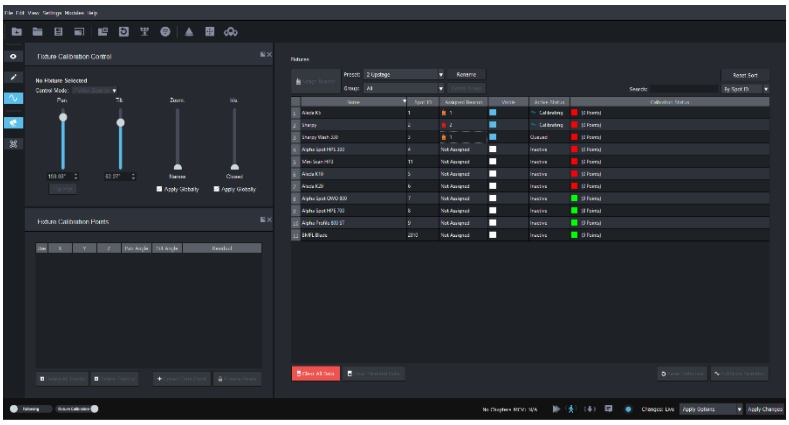


Figura 9 - Vista de calibração de robótica do Blacktrax

A taxa de atualização do sistema fica limitada principalmente pela velocidade de captação das câmaras, 100fps[4].

Devem ainda ser tomadas algumas precauções adicionais de extrema importância para o bom funcionamento do sistema, como a certificação de que: o espaço a monitorar não tem qualquer incidência de luz solar; não existem materiais refletores de luz Infravermelha, visto que pode refletir o padrão dos led's; não existem fontes de infravermelhos como alguns projetores de luz ou mesmo fogo. As câmaras têm de ser colocadas em locais estacionários sem qualquer variação no espaço ou a precisão irá alterar-se de forma imprevisível.

[4] Frames por segundo. Indica a taxa de atualização.

2.3.1.3 Interface gráfica de utilizador

Existem três softwares necessários para o sistema funcionar em pleno. O **Motive** serve para gerenciar e calibrar todas as BTCameras. O **Blacktrax** é o software de programação das memórias de rastreamento. O seu modo de programação consiste em *chapters*, ou seja, o conjunto de relações entre vários led's e determinado número de robôs que os perseguem e, por *books*, que são o conjunto desses *chapters*.

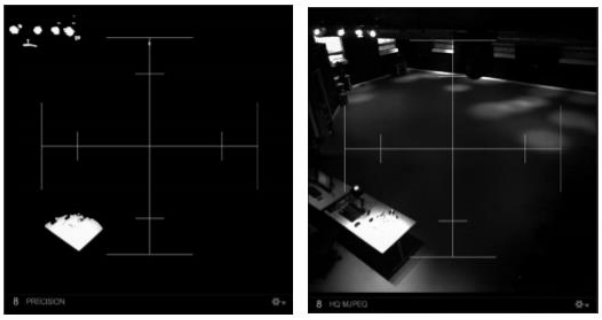


Figura 10 - Modo de tracking à esquerda e Modo de calibração à direita

A interface gráfica está dividida em três: *Calibration View* que possui ferramentas de calibração; *Edit View,* que possibilita a programação de *chapters*; *Live View*, que possibilita o monitoramento ao vivo da performance.

Através da criação de áreas existe ainda a possibilidade de ativar atributos de modo a que a abertura do feixe de luz ou a focagem seja sempre o mesmo independentemente da distância dos led's.

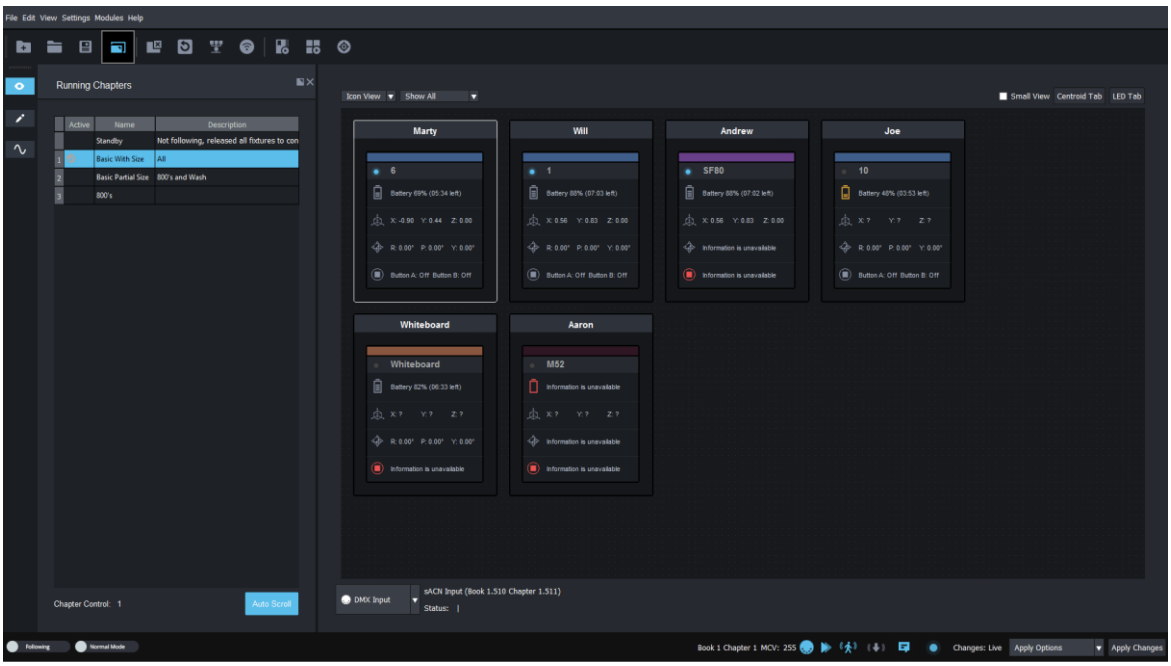


Figura 11 - Exemplo da Live View

O **BTWYSIWYG é** o software gémeo do WYSIWYG mas com funcionalidades extra para interação com a rede Blacktrax. Com este software, além de se poder conceber todo o desenho de luz também se pode importar e fazer o patch de elementos rastreáveis como os leds's dos BTBeacons, e as BTCameras.

A comunicação com a mesa de luz é unidirecional. A mesa de luz envia DMX através de Art-net ou sACN para o sistema Blacktrax que, por sua vez processa o sinal e envia posteriormente para a robótica. Para que o Blacktrax consiga efetuar a perseguição de luz é necessário que o patch dos robôs utilizados seja igual ao patch do Blacktrax. A partir da mesa de luz é então possível controlar duas funções do Blacktax através do controlo de canais DMX: ativar o sistema e selecionar o capítulo ativo.

2.3.2 Zactrack

Este é um sistema de *tracking* que migrou há cerca de um ano para a tecnologia UWB. Apesar da reduzida documentação foi possível detalhar algumas das características e funcionalidades deste sistema.

2.3.2.1 Estrutura e funcionamento

O sistema é constituído por um conjunto de âncoras, do tamanho de routers domésticos, e *tags* que são colocadas nos indivíduos a rastrear. Tecnicamente, a quantidade mínima é de 4 mas o recomendado é um mínimo de 6 âncoras.

Figura 12 - Âncoras e tags do Zactrack

A distancia máxima global do sistema atual é de 100 metros e possui uma taxa de atualização máxima abaixo dos 40Hz.

Em comunicação com as âncoras através de fibra ótica está o **Zacktrack Core** que trata de calcular a posição das *tags* e de gerir toda a parte de controlo. Esta é uma unidade baseada em Linux que permite, tal como o Blacktrax, a integração de luz, som, vídeo e controlo de dispositivos como a orientação de câmaras de vídeo.

A integração de luz é feita autonomamente pelo **Zactrack Light Control** ou, em combinação com uma mesa de luz, através da comunicação via Art-net, ACN ou DMX. Na documentação da versão de sistema anterior, esse software permite o controlo de atributos como a iris, *dimmer*, *zoom* e *focus* e permite ainda a criação de zonas que permitem o controlo diferenciado desses atributos, tal como faz o Blacktrax. Possui ainda a funcionalidade **Dynamic effects** que

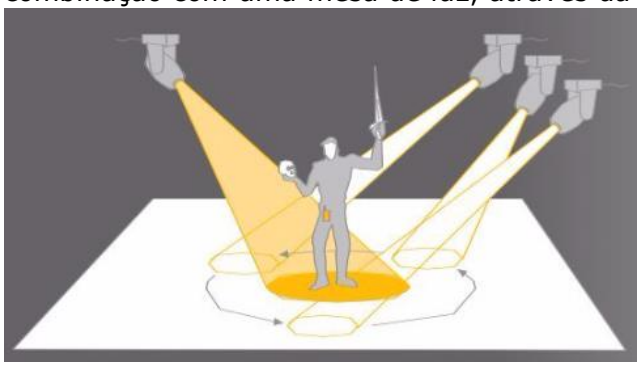


Figura 13 - Exemplo de Dynamic effects do Zactrack

permite o uso de alguns efeitos dinâmicos à volta de uma *tag*, como por exemplo o efeito circular de planetas a girar à volta do sol como mostra a figura ao lado.

Incorpora também o **Zacview**, com um conceito semelhante ao BTWYSIWYG, que permite a visualização e criação do espaço tridimensional e todo o seu conteúdo de iluminação, cenário e outros. É ainda possível a visualização em tempo-real de movimentações de equipamento cénico e das *tags*.

Para efetuar a calibração é necessário a instalação do sistema no local e iniciar o modo de calibração. Nesse modo, o sistema calibra-se autonomamente, não necessitando do conhecimento prévio das condições e características do espaço ou de intervenção humana.

2.3.3 Outros sistemas

Tal como o Blacktrax, o Coolux baseia-se na tecnologia de luz infravermelha modulada. Uma das vantagens deste em relação ao rival Blacktrax é a elevada taxa de atualização, podendo mesmo chegar aos 250fps em comparação com os 100fps do Blacktrax.

Tanto o Vicon como o Optitrack ou o Arttrack são também sistemas óticos de rastreamento como o Blacktrax mas são destinados a ambientes controlados para modulação de personagens virtuais nas áreas da animação digital e cinema ou por exemplo para estudos biomecânicos.

O Vicon é um sistema de captura de movimentos bastante popular e como muitos outros, funciona através de luz infravermelha. Tal como o Blacktrax, são necessárias condições bastante peculiares para uma resposta otimizada do sistema como a inexistência de luz solar e a inexistência de materiais refletores de luz infravermelha desse espetro. Este sistema utiliza *motion suits* que são fatos dotados de múltiplas esferas chamados marcadores. Como é um sistema ótico passivo, os marcadores destes fatos são constituídos por um material que reflete a luz infravermelha que, ao serem bombardeados por luz infravermelha o software deteta e analisa de forma a constituir um exoesqueleto virtual.

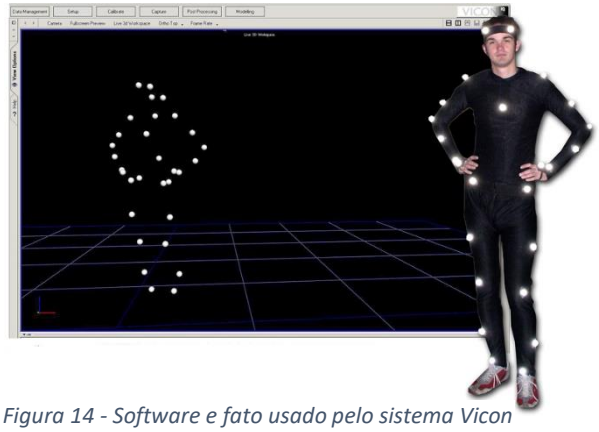


Figura 14 - Software e fato usado pelo sistema Vicon

2.3.4 Suporte à interação de luz

Os sistemas profissionais de rastreamento para artes performativas já existem há algum tempo, mas só permitem uma limitada interação com a luz, como por exemplo o Zactrack com os seus *dynamic effects*. Tanto esse como o Blacktrax possuem protocolos de comunicação para que aplicações externas possam aceder à informação de rastreamento disponível pelo servidor, mas na investigação efetuada não foi possível identificar qualquer aplicação que já tenha utilizado essas vias de comunicação com o intuito de interação com a iluminação.

De facto, na maioria das performances, a utilização do rastreamento da posição, gestos ou outros para interagir com o resto da iluminação presente, é utilizada na produção de conteúdos de vídeo ao vivo. Na maior parte delas é possível verificar o uso da Kinect, mas existem alternativas como o "Xtion Pro LIVE" da Asus, ou o "RealSense" da Intel, cujas funcionalidades e características são semelhantes.

O Kinect capta gestos precisos a uma distância relativamente curta. Das tecnologias apresentadas, este é um típico dispositivo de *Computer Vision* no qual recolhe e processa dados visuais para reconhecimento facial, gestual e de voz. Ao incluir um kit de desenvolvedor com bibliotecas *open-source*, este dispositivo foi adaptado para todo o tipo de aplicações dentro e fora das artes performativas.

Existem alguns projetos como o "Pattern Recognition", de Memo Akten, que utiliza pelo menos duas kinects que determinam padrões do corpo que servem de gatilho para a interação com a robótica de iluminação.

Este dispositivo possui uma câmara com resolução 1280x960, um sensor de profundidade por infravermelhos e quatro microfones. Com este dispositivo é possível detetar até 6 pessoas e 25 articulações por pessoa.

No entanto, o alcance de 4,5 metros, a taxa de atualização máxima de 30fps e o ângulo de visão de 70x60 graus, adicionado às necessidades extra para interligação de mais dispositivos, tornam este dispositivo pouco indicado para o rastreamento interativo em performances de média e larga escala.

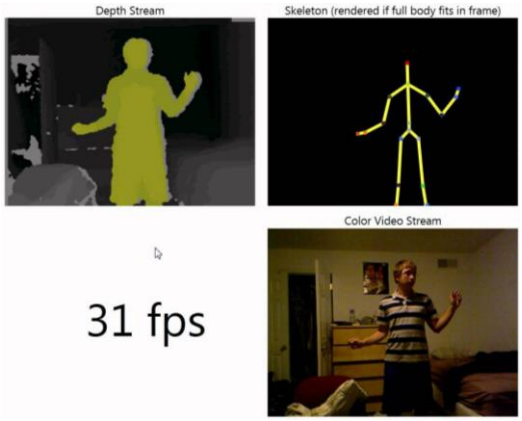


Figura 15 - Deteção de corpo e articulações, captação de imagens RGB e frequência de atualização da Kinect

2.3.5 Diferenciação de sistemas

Cada um dos sistemas referidos possui prós e contras quando comparados com outros e como tal, foi elaborada uma tabela de comparação dos principais sistemas falados para melhor compreender as características de cada um. O Kinect não foi sequer comparado devido à falta de capacidades técnicas óbvias para o tipo de aplicações que este projeto se debruça.

Sistema	Tecnologia	Taxa Atualização[5]	Indivíduos	Apto para iluminação	Preço[6]
Blacktrax	IR Ativo	100 fps	< 85	Sim	>80.000 €
Zactrack	UWB	< 40 Hz	< 20	Sim	>45.000 €
Vicon	IR Passivo	< 250 fps	V. ilimitado	Não	>50.000 €

Tabela 2 - Comparação de principais sistemas

Todos os três sistemas possuem alcances dinâmicos, utilizam marcadores e têm precisão bastante semelhante.

O Vicon possui várias características negativas como a inclusão de fatos de rastreamento e a falta de suporte nativo à iluminação performativa, não sendo à partida sequer posto como uma opção no mercado de iluminação.

A taxa de atualização comparada é bastante importante. A leitura errónea de dados é provocada pela baixa taxa de atualização, enquanto que um valor elevado indica uma aproximação otimizada da posição real, e possibilita o uso otimizado de funções matemáticas que aumentam o seu rigor.

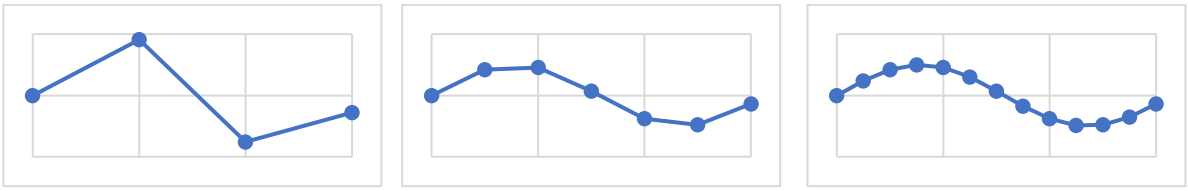


Gráfico 1 - Exemplo de medição do mesmo trajeto em 2D com 4, 7 e 13 amostras na mesma unidade de tempo

A questão aqui seria resolvida se soubéssemos qual a taxa mínima de atualização para que o olho humano não detete esses pontos de amostragem. O objetivo é que uma perseguição de luz seja suave e não se veja o feixe de luz a saltar de ponto em ponto. Por exemplo, no cinema utiliza-se o padrão 25/30fps, a luz elétrica é emitida a 50/60Hz e nas televisões mais recentes pode verificar-se valores superiores a 240Hz.

[5] Amostras por unidade de tempo (1fps = 1Hz)
[6] Preços tabelados, podem ser sujeitos a negociação

Na perseguição de luz, a taxa de atualização mínima do sistema de rastreamento deve ser calculada com base em diversos fatores, como a velocidade do que vai ser rastreado. Nesse caso, existe a possibilidade de performers correrem ou mesmo andar em bicicletas. Este campo não é de todo uma ciência exata, pelo que consideramos que quanto mais elevada for essa taxa, mais fiabilidade se alcança.

A quantidade máxima de indivíduos que podem ser rastreados é também um fator de grande importância, pelo que o Blacktrax possui uma capacidade de rastreamento bastante superior em relação ao Zactrack, até porque cada um dos 85 indivíduos pode ter até 3 stringers rastreáveis, ou seja, podem ser individualmente rastreados 255 pontos. No entanto, o Zactrack vinga pelo preço, que é quase metade do Blacktrax, tornando este sistema indicado para pequenas e médias performances.

3. Desenvolvimento do sistema

3.1 Metodologia

Neste projeto foi essencialmente utilizada uma metodologia de prototipagem que é muitas vezes utilizada em projetos de engenharia. Em projetos deste género, existem essencialmente três fases que são realizadas de forma repetitiva e em paralelo: análise, desenvolvimento e implementação. Estes ciclos ocorrem devido á experimentação constante do produto que está a ser desenvolvido. Ao criar uma nova parte de um protótipo, é necessário implementa-la e analisar os novos resultados para validar esse novo desenvolvimento, só parando quando o protótipo alcança os objetivos pretendidos com as melhores funcionalidades possíveis, passando a produto final.

Neste caso, procedeu-se ao desenvolvimento inicial de pequenos testes que iriam validar as várias técnicas e ferramentas que se pretendiam utilizar. Só após o sucesso destes se avançou para a fase de prototipagem na qual se integrava tanto os módulos de *software* como *hardware*. Com este método, uma determinada funcionalidade do sistema nunca é verdadeiramente definida na sua criação, mas sim ao longo do desenvolvimento geral do sistema. Algo que no início parece o mais adequado para determinada situação, pode afinal tornar-se incompatível com outras características do protótipo ou mesmo ficar obsoleto.

Com esta ideia em mente, determinou-se um desenvolvimento encapsulado das várias partes do sistema, de forma a que a alteração, adição ou remoção dessas partes fosse o mais cirúrgico possível de forma a não alterar o resto dos desenvolvimentos. Desta forma foi possível moldar esses módulos múltiplas vezes de forma a encontrar as melhores relações entre a performance, funcionalidades e a interação com o utilizador.

3.2 Pozyx

No final de 2016, durante a pesquisa de tecnologias e técnicas de rastreamento que pudessem servir o objetivo do projeto, surgiu o Pozyx. O Pozyx é uma solução de prototipagem para Arduino que utiliza a tecnologia UWB para obter a localização através do método da trilateração. Este é um sistema bastante similar ao Zactrack, e é baseado em âncoras e *tags*. Na versão 1.0, a frequência de atualização deste sistema era de cerca de 40Hz para uma *tag*. Sendo essa frequência diretamente proporcional ao número de *tags* existentes, um sistema com quatro dispositivos teria uma atualização de 10Hz por cada *tag*, o que tornava o sistema inviável para o rastreamento rápido de pessoas ou objetos, pelo menos em pequena e média escala. Em abril de 2017, após o lançamento da versão de *firmware* 1.1, as características gerais do sistema sofreram algumas alterações,

tornando o sistema mais apelativo para o objetivo deste projeto, como é explicado abaixo, e tornando-o no ponto de partida para a escolha das ferramentas e técnicas usadas.

3.2.1 Especificações

A versão de *firmware* 1.1 incorporado no sistema possibilita as seguintes funcionalidades:

- Precisão inferior a 10 centímetros.
- Frequência de atualização do posicionamento de até 140Hz.
- Frequência de atualização da medição de distância entre dois dispositivos de até 350Hz.
- Alcance de medição da distância entre dois dispositivos até 100 metros sem obstáculos.
- Alcance de posicionamento até 120m^2 com o pack de 4 âncoras.
- Algoritmos nativos de filtragem e suavização de dados de posicionamento.
- Posicionamento da *tag* não necessita de linha de vista para as âncoras.
- Médio/baixo consumo energético.
- Bibliotecas de integração de código aberto.
- Sem interferências com Wi-Fi, Bluetooth ou praticamente qualquer outra tecnologia de radiofrequência.

Figura 16 - Pack Pozyx de 4 âncoras e 1 tag

- Dimensões das *tags* relativamente reduzidas (68 X 53 mm).
- Custo do pack de 4 âncoras e 1 *tag* inferior a 600€.

Para a configuração e otimização da localização o sistema providencia quatro parâmetros relativos à própria tecnologia UWB:

- Canais UWB: são providenciados 6 canais individuais, ou seja, a comunicação entre dispositivos de canais diferentes é impossível e sem interferências entre eles.
- *Bitrate*: possibilita velocidades de comunicação de 110kbit/s, 850kbit/s ou 6,81Mbit/s. Quanto maior a velocidade, menor o tamanho das mensagens e alcance do sistema.
- *Pulse Repetition Frequency* (PRF): que indica a frequência com que os pulsos são emitidos. Pode ser de 16 ou 64MHz e tem muito pouco impacto nas velocidades e

alcance de comunicação, no entanto, quanto maior for mais rápida é a taxa de atualização.

— *Preamble Length*: que significa comprimento de preambulo, possibilita a escolha de 7 modos, 4096, 2048, 1024, 512, 256, 128 ou 64 símbolos. Um pacote de dados transmitido é dividido em várias partes. Um preambulo é a primeira parte de um pacote de dados e serve para sincronizar timings de transmissão entre emissor/recetor. Uma pequena dimensão do preambulo significa que serão transmitidos menos dados e como tal a velocidade de comunicação aumenta enquanto o alcance diminui.

Com a ajuda de gráficos experimentais oficiais, é possível perceber que a melhor frequência de atualização é obtida com um preâmbulo de 64 símbolos e um *bitrate* de 6.81MHz. Selecionada a melhor frequência de atualização o alcance será sem dúvida bastante inferior aos 120m^2, que equivale aproximadamente a um espaço de 11 metros por 11 metros. Apesar disso, como o sistema é escalável, é possível adicionar mais âncoras para que o alcance seja superior a esse limite.

3.2.2 Calibração e funcionamento

Pode dizer-se que a instalação e uso do Pozyx é efetuada de forma mais rápida que sistemas como o Blacktrax, que se desenvolve ao longo de 5 passos:

1. Posicionamento das âncoras: as âncoras devem ser posicionadas à volta da área de rastreamento, e nunca em linha, de forma a otimizar o método de trilateração.

2. Medição da distancia bidimensional das âncoras a um ponto de referência comum: Utilizando um qualquer ponto do espaço onde se encontram as âncoras, medir as distâncias cartesianas (x ,y e z) dessa referência. Geralmente utiliza-se a posição de uma das âncoras como referência para facilitar as medições.

3. Ligar as âncoras à corrente elétrica: Todos os dispositivos necessitam de 5V em corrente contínua e o consumo de cada dispositivo nunca chega a 10W.

4. Ligar uma das *tags* ao computador via USB: Uma das *tags* deve ficar perto do computador que vai utilizar as bibliotecas e programas cedidos para processar a localização das *tags*.

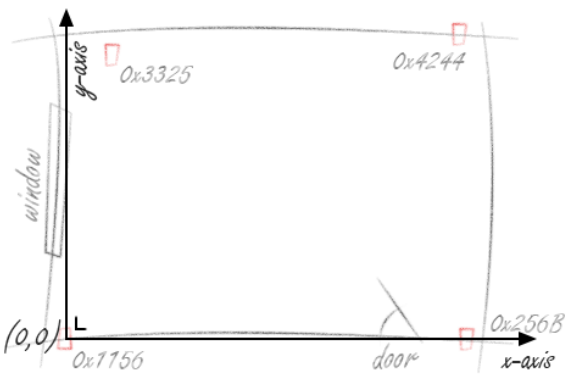


Figura 17 - Definição de uma âncora como ponto de referência. Âncoras representadas a vermelho

5. Correr os programas cedidos para obter dados de localização e orientação: após ligar todo o sistema é necessário selecionar

o modo de funcionamento e inserir as coordenadas das âncoras para visualizar a localização e orientação das *tags*.

Apesar de estar já pensado um modo de calibração automático, o sistema ainda necessita da posição das âncoras pré-definida pelo utilizador para identificar e localizar as *tags*. Com esta funcionalidade ativa numa próxima versão, o ponto dois poderá ser eliminado simplificando toda a calibração.

Para esboçar o limite máximo deste sistema, considerou-se uma taxa de atualização mínima igual ao padrão do protocolo DMX, 44 Hz e a frequência máxima de 140Hz. Chegou-se à conclusão que o limite máximo de *tags* em simultâneo, contando com margens de erro de medições e latência do sistema a nível global, seria constituído por 6 grupos (canais) de 2 conjuntos (PRF) de 4 âncoras e 2 *tags*, ou seja, com 48 âncoras, é teoricamente possível o rastreamento de 24 *tags* a uma velocidade aproximada de 70Hz com a máxima precisão num espaço de cerca de 7 x 7 metros (estimativa pessoal).

3.2.3 Vantagens e desvantagens

À partida, este conjunto de características tornam o Pozyx numa excelente ferramenta de rastreamento viável e de baixo custo. Tal como o Zactrack, e ao contrário do Blacktrax, não existe necessidade da existência de uma linha de vista desobstruída com as âncoras, facilitando a invisibilidade das *tags* e possibilitando um ancoramento diversificado, prático e de fácil instalação. A sua fraca interferência com outro tipo de radiofrequências torna este sistema bastante fácil de calibrar e com uma fiabilidade bastante aceitável.

No entanto, existe um grande inconveniente, comum a qualquer tecnologia de radiofrequência, a sensibilidade a materiais metálicos. Sendo as ondas de rádio formadas por campos eletromagnéticos, e o metal um excelente condutor de eletricidade e magnetismo, o que acontece é que o metal absorve essas ondas, provocando distúrbios nas características das ondas e consequentemente nas medições. Com isto, as âncoras não devem ser posicionadas a menos de 30cm de materiais metálicos, segundo a documentação. A reação óbvia seria desenvolver suportes de material não metálico para resolver o problema.

O preço, a característica *open-source*, as taxas de atualização de até 140Hz, a possibilidade escalável do sistema e as diversas facilidades de instalação, calibração e funcionamento podem elevar este sistema ao mesmo patamar do Blacktrax ou do Zactrack.

3.3 Introdução à prototipagem

Antes de começar a elaborar qualquer parte deste projeto, foi necessário determinar as técnicas e ferramentas necessárias para o desenvolvimento de um protótipo de rastreamento funcional. Assim sendo, antes de proceder ao desenvolvimento do sistema foi efetuada a escolha de ferramentas seguida da planificação de software.

3.3.1 Escolha de ferramentas

Compreender as possibilidades existentes e escolher as melhores ferramentas para a obtenção de um sistema simples e consistente é fundamental para o sucesso de qualquer projeto. Assim, dado o conhecimento pessoal de algumas linguagens de programação, o C++ foi selecionado como a linguagem que melhor se adequava. Além de ser uma linguagem de programação de alto nível, conta com um vasto suporte online e um amplo conjunto de ferramentas de desenvolvimento em múltipla plataforma.

Infelizmente as bibliotecas do Pozyx em C++ são destinadas apenas à plataforma de prototipagem Arduino pelo que, alterar essas bibliotecas para serem utilizadas em qualquer sistema operativo levaria demasiado tempo, optando assim pela utilização das bibliotecas em Python que seriam à partida facilmente incorporadas no projeto.

Assim sendo, o software responsável por todo o sistema e comunicações seria escrito em C++ com a possibilidade futura de poder correr em ambientes alternativos ao Windows como o Linux e o Mac OS.

Após a linguagem de programação base, foi necessário escolher quais as bibliotecas de desenvolvimento gráfico aptas para a integração com C++. Inicialmente foi considerado o OpenFrameworks (ofx), que no fundo é um conjunto de ferramentas em C++, de código aberto, que possibilita vastas opções gráficas desde botões, controlos, gráficos, vistas, janelas, entre outros. Foram ainda consideradas algumas opções como o MFC ou GTK+ mas foram facilmente descartadas pela complexidade de integração. No entanto, surgiu o Qt, uma plataforma com capacidades extra bastante apelativas como é explicado abaixo.

Qt

O Qt possui um editor de código em C++ e um modo de desenho para criação de interfaces gráficas com funcionalidades drag-n-drop. Esta plataforma permite ao utilizador uma integração detalhada da interface de utilizador com o código fonte.

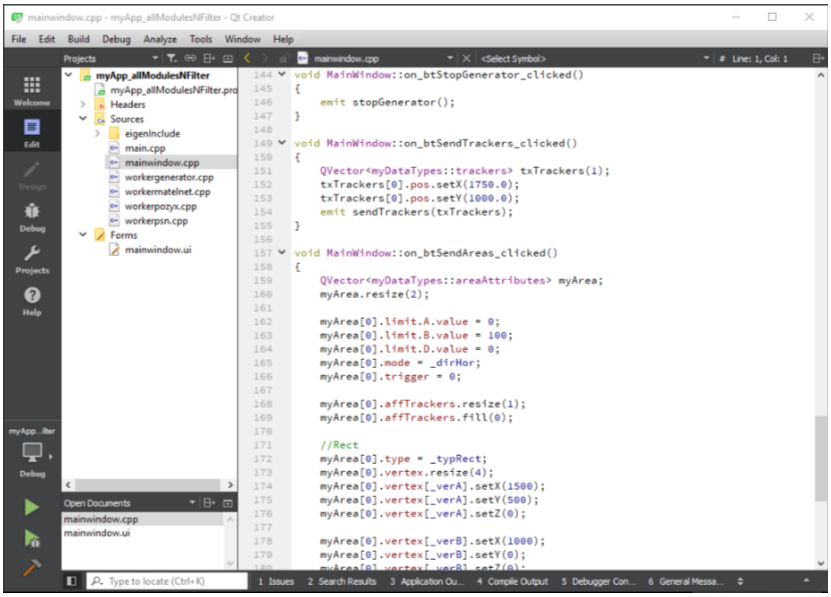


Figura 18 - Vista de editor de código do Qt

Ainda que nesta fase a interface gráfica não fosse uma prioridade, a possibilidade de a deixar integrada com o resto do código sem necessitar de alterar o código de forma profunda foi uma das razões para a seleção desta plataforma. Uma outra vantagem é a inclusão de ferramentas simples que permitem o estilo de programação *multi-thread*, ou seja, a criação de módulos que correm paralelamente dentro do mesmo software.

PosiStageNet

A forma como se processa os dados de localização para a sua perseguição de luz e, a mesa de luz a utilizar, foram também determinadas devido à existência de um protocolo na mesa de luz grandMA2[7], o *PosiStageNet* (PSN). O PSN é um protocolo de código aberto de transmissão de dados de localização e orientação. Este protocolo é destinado a qualquer área que necessite do rastreamento de algo ou alguém. Na grandMA2 o protocolo é usado unicamente para a perseguição de luz na qual podemos associar os dados dos *Trackers*[8] recebidos a *Stage Markers*[9]. Assim, segundo a documentação, para adicionar um ponto de rastreamento de luz é necessário seguir os passos:

1. Adicionar um *stage Marker* ao patch existente na mesa;
2. Criar o desenho de luz tridimensional no software MA3D ou na própria mesa de luz;
3. Ativar o protocolo PSN;
4. Atribuir a ID do *stage marker* a qualquer *Tracker* existente;
5. Selecionar a robótica pretendida para fazer a perseguição;
6. No *encoder Position*, mudar do modo Pan/Tilt para "XYZ";

[7] Utilizada a versão 3.3.2.2 da grandMA2
[8] Nomenclatura técnica do PSN que se refere a pontos de rastreamento como *beacons* e *tags*
[9] Nomenclatura técnica da MA2 que se refere à representação tridimensional de *Trackers*

7. Atribuir o *stage marker* ao atributo *marker*;

A existência deste protocolo na MA2 possibilita a passagem de todos os cálculos necessários à perseguição para a própria mesa de luz, de tal forma que no software a desenvolver apenas seria necessário transformar os dados de posicionamento e orientação das *tags* em PSN. Existe uma biblioteca oficial *open-source* de PSN escrita em C++, munida de algumas aplicações exemplo, o que facilita ainda mais o desenvolvimento desta funcionalidade.

Controlo de interação

De forma a criar interações com a iluminação conforme a posição do objeto ou pessoa, pensou-se inicialmente na inclusão de um modo de interação em que uma dada área iria ter impacto nos canais DMX predefinidos pelo utilizador. Esta ideia foi rapidamente esquecida porque para isso seria necessário ter uma espécie de patch que facilitasse a seleção desses canais e a inclusão de uma biblioteca Art-net ou sACN, por exemplo, o que de momento iria retirar muito tempo ao projeto. Em vez disso, surgiu a ideia de interligar o software à programação de luz existente na mesa o que implicou procurar um meio de comunicação com esta.

Com a oportunidade de por à prova a programação convencional de luz de forma a incluir métodos de interação com a localização de algo ou alguém, surgiram as possibilidades *MA-Net2*, *Remotes* e *Telnet* sendo que a primeira foi descartada à partida por ser uma via fechada na qual apenas dispositivos certificados pela MA Lighting podem ter e utilizar.

A opção *Remotes*, é a via oficial de controlo remoto da mesa, acessível a partir de qualquer navegador de internet que se encontre na mesma rede. O funcionamento deste método baseia-se num servidor web, alocado na grandma2 que, quando acedido providencia uma interface gráfica que fornece um conjunto limitado de controlos semelhante ao que existe na mesa de luz física.

A opção *Telnet* é semelhante à anterior, mas sem a parte gráfica. É uma via de acesso direto à linha de comandos da grandMA2 através do protocolo Telnet que faz uso de um outro, o *Transmission Control Protocol* (TCP), como meio de transporte. Este é um protocolo padrão de internet que permite a interface de terminais e aplicações através de uma rede. Fornece funcionalidades básicas para criar uma via de comunicação entre cliente e servidor através de texto interativo. Neste caso, a grandMA2 possui um servidor *Telnet* que providencia acesso a todos os comandos possíveis que existem na documentação, ou seja, é possível o controlo total da mesa.

Na internet estão disponibilizadas inúmeras bibliotecas Telnet em C++, o que facilita a integração deste protocolo de comunicação em aplicações de terceiros.

3.3.2 Planificação de software

Desde o início que o software foi pensado de forma modular para que a resolução de problemas e a integração tardia de outros módulos fosse o menos complexa possível e de simples resolução. Além disso, o método de operação foi pensado de modo semelhante à programação de uma mesa de luz, que neste caso seria um conjunto de *Layers*, mais tarde controlados através de Art-net, que contêm áreas de interação cuja função seria armazenar os dados dos controlos da MA2 associados para depois reagirem conforme as *tags* se encontrassem dentro ou fora das áreas. Foi definido desde início que a GUI seria um módulo que só iria ser desenvolvido se realmente existisse tempo para isso. No entanto, a programação de todo o software foi desde sempre elaborada de forma a integrar essa parte mais tarde.

3.3.3 Gestão de Projeto

Após a escolha de ferramentas e técnicas adequadas para o projeto, foi altura de planear o desenvolvimento do projeto com vista a apresentação do sistema como um protótipo de produto final. Assim sendo, foram definidas algumas datas com alguma margem de erro incluída, sendo elaborada uma tabela compacta para gestão de tempo pessoal:

Até	Descrição
30.Fev	Testes iniciais e desenvolvimento inicial do sistema
30.Mar	Completado desenvolvimento de software - PSN
30.Abr	Completado desenvolvimento de software - Interação
5.Mai	Testes práticos
15.Mai	Consolidação de software
15.Jun	Desenvolvimento de software – GUI
20.Jun	Testes práticos
30.Jun	Consolidação de software
5.Jul	Ensaios
10.Jul	Performance

Tabela 3 - Planificação das diversas fases do projeto

3.4 Desenvolvimento de software

3.4.1 Introdução ao Qt

O Qt é uma *framework* de criação de aplicações multi-plataforma, muitas vezes escolhida pelo seu conjunto de ferramentas gráficas. Possui bibliotecas próprias que se assemelham às nativas da linguagem C++ mas cujas funcionalidades facilitam a implementação.

O Qt introduz uma funcionalidade única de comunicação baseada em *SIGNALS* e *SLOTS*, ou seja, em eventos sinalizados a partir de "*SIGNALS*" que são tratados nas *SLOTS*. Dando um exemplo: se no modo gráfico se desenhar um botão, este automaticamente possui uma lista de *SIGNALS* como *clicked()*, *pressed()*, *released()*, entre outros. Este *SIGNAL* é uma função, que pode transportar variáveis para uma ou várias funções chamadas de *SLOTS*. *SLOT* é uma função comum com a particularidade de poder ser chamada por *SIGNALS* de objetos externos ao que se insere. Neste exemplo, para definirmos o que o botão faria se fosse clicado, apenas seria necessário interligar o *SIGNAL clicked()* do botão à sua *SLOT* predefinida e aí programar o que se pretendesse.

Uma outra funcionalidade, de certa forma "otimizada" é o *multithreading*, ou seja, a capacidade de criar módulos independentes a correr em paralelo com comunicação bidirecional.

Essas duas funcionalidades, fizeram com que o desenvolvimento de todo o projeto fosse bastante mais rápido que o esperado.

3.4.2 Desenvolvimento inicial

Após ter definido o POZYX como sistema base do projeto e o Qt como editor de código e interface gráfica padrão do software foi necessário identificar as partes mais complexas ou árduas de programar de forma a prioriza-las. Foi criado o primeiro esboço para compreender a as comunicações mínimas necessárias.

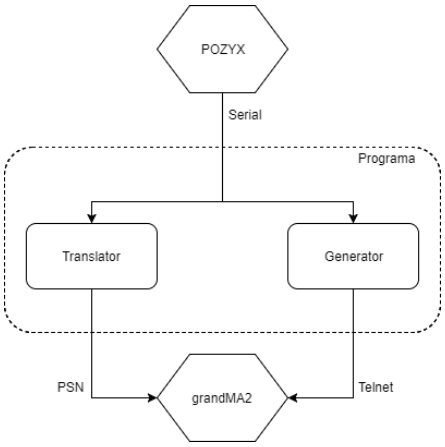


Diagrama 2 - Esboço inicial do software

Numa primeira análise, os dados do Pozyx seriam transmitidos ao programa e aí seriam partilhados por dois módulos. Um deles transformava diretamente esses dados no protocolo PSN. O outro módulo trataria da parte de interação conforme a posição e posteriormente controlava a mesa através de *telnet*. Ambos os protocolos seriam enviados para a mesa via *ethernet*.

Com isto, ficaram pendentes vários pontos importantes como:

1. localização do software e de que forma se comunica com a mesa;
2. qual a via de comunicação entre o programa em *Python* do Pozyx, e o software a desenvolver;
3. de que forma o módulo de interação vai processar os dados de localização com a iluminação;

Tendo em conta que o sistema é vocacionado para artes performativas é percetível que o software que interage com o Pozyx deveria ficar na régie e não no palco, porque é um meio de controlo do sistema. Como tal, surgiu a possibilidade de inserir um microcomputador junto a uma das âncoras, o Raspberry Pi 3, que fizesse o processamento do programa em *Python* e enviasse os dados de localização e orientação para a régie através do protocolo TCP.

Raspberry Pi 3

O raspberry Pi 3 é uma unidade de processamento de dimensões reduzidas, um micro computador. Apesar do seu baixo poder de processamento esta é uma escolha bastante comum para prototipagem pela sua facilidade de utilização, dimensões e principalmente pelos seus 26 multifuncionais *General Purpose Input/Output* (GPIO). Os sistemas operativos suportados são inúmeros mas o mais usado é sem dúvida o Raspbian Linux, uma versão Linux[10] criada para obtenção da máxima compatibilidade com este dispositivo.

Sendo que a aplicação em Python do POZYX possui especificações bastante reduzidas, o Raspberry Pi pareceu uma boa aposta como plataforma de comunicação entre o sistema, que possivelmente seria localizado num palco e o software a desenvolver, que se localizaria numa régie. Com isto, apenas seria necessário integrar o protocolo TCP para comunicação com a régie.

[10] Sistema operativo de código aberto

Protocolo TCP

O *Transmission Control Protocol* é um dos protocolos mais usados pela internet e é geralmente chamado de TCP/IP. Este é um protocolo bastante utilizado pela sua confiabilidade na medida em que possui um sistema de verificação de dados para que não haja erros no seu envio/receção. Este é um protocolo de transporte sobre o qual assentam outros protocolos como o SSH, HTTP ou mesmo o Telnet.

Este protocolo usa a comunicação através do método servidor/cliente, no qual se um cliente se desligar do servidor, necessita de um novo pedido de ligação ao servidor. Este protocolo está já integrado em muitas bibliotecas facilitando a integração num programa em *Python* ou C++.

Posto isto, os dois primeiros pontos ficariam resolvidos, restando apenas a questão do processamento de interações. Uma das primeiras opções foi a criação de áreas de interação. Se um individuo com uma *tag* entrasse numa dessas áreas iria ativar um ou vários controlos na mesa de luz. Foram inicialmente listadas duas opções atribuíveis a uma área:

- Direção de interação: horizontal, vertical, centro ou *flat*;
- Margens: Limite A, B ou *Default*;

A direção determinaria por exemplo se um *fader* da grandMA2 subia à medida que o individuo se movia da esquerda para a direita, verticalmente, ou se ao entrar na área ativava um outro controlo como uma *Macro*, um botão de um *Executor* ou qualquer outro.

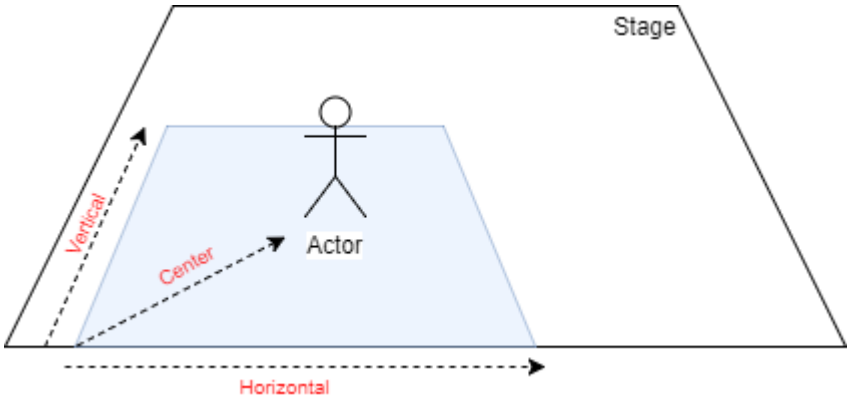


Ilustração 2 - Exemplificação das possíveis direções

Já as margens teriam que ver com a sensibilidade da interação. Por exemplo, ao entrar numa área poderíamos iniciar a subida de um *fader* apenas a partir de uma certa distancia definida por esta variável. A margem *default* seria o valor com que esse *fader* deveria ficar caso o individuo saísse da área a meio de uma interação.

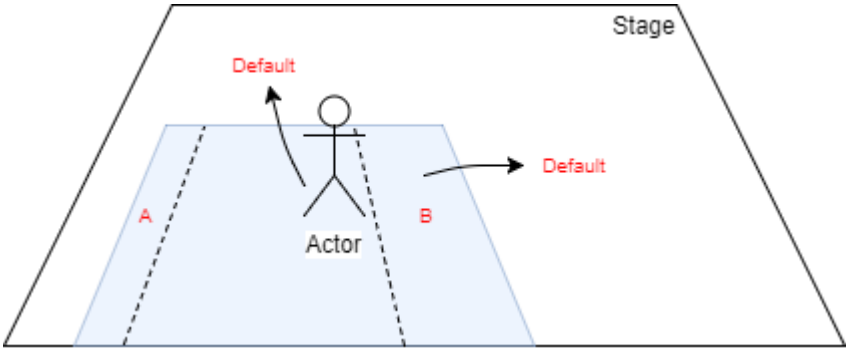


Ilustração 3 - Exemplificação das possíveis margens

Após consolidar estes pontos foi elaborado um outro esboço com as novas funcionalidades e necessidades:

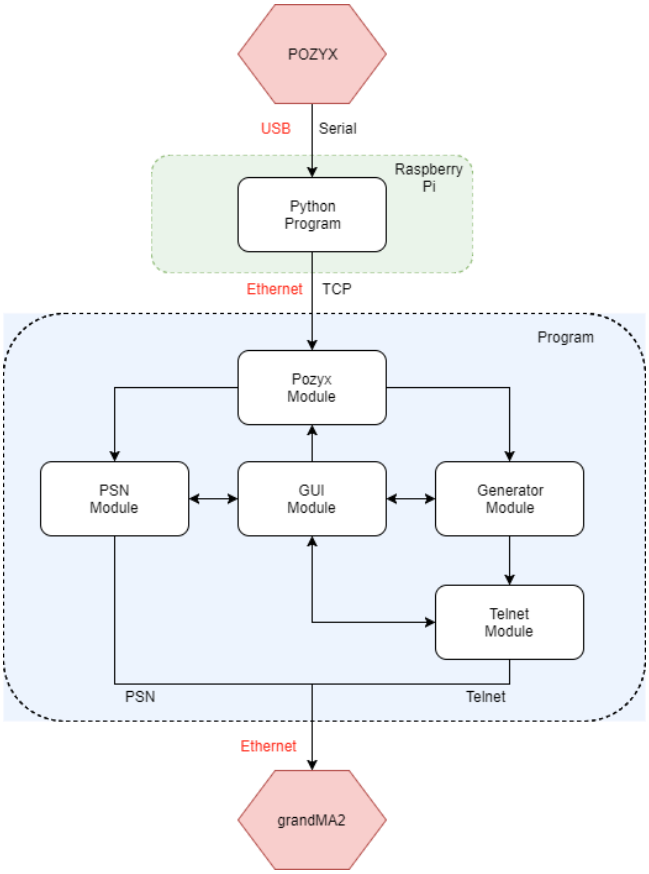


Diagrama 3 - Segundo diagrama de funcionamento do software

Ainda antes da chegada do Pozyx foram feitos testes iniciais. Para cada parte do software foi elaborado um programa de complexidade básica para testar essas funcionalidades de forma individual:

- Teste de comunicação via *Telnet*

Após várias tentativas de integração de bibliotecas de *Telnet* encontradas na internet, percebeu-se que estas eram bibliotecas para criação gráfica de um terminal *Telnet.* Visto que a ligação á mesa de luz se iria efetuar de forma invisível para o utilizador, preferiu-se utilizar uma aproximação simplificada do protocolo de transporte do *Telnet*, o TCP, utilizando assim as bibliotecas nativas do Qt para tal.

Foi então criado um programa de teste, em linha de comandos, que quando executado enviava automaticamente comandos predefinidos para a grandMA2. Abaixo está a maior e mais importante parte desse programa. A explicação de cada parte pode ser encontrada à direita da mesma[11].

```
#include "socketize.h"                          // Inclusão de ficheiro complementar onde se regista funções
                                                //    e variáveis criadas peloprogramador
Socketize::Socketize(QObject *parent) : QObject(parent) // Função de inicialização sem parâmetros.
{
}

void Socketize::work()                          // Função principal que faz a ligação e envia comandos
{
    socket = new QTcpSocket(this);              // Criação de um novo Socket
    socket->connectToHost("127.0.0.1",30000);   // Criação da ligação com a grandMA2
                                                //   IP da grandMA2: 127.0.0.1; Porta Telnet na grandMA2: 30000
    if(socket->waitForConnected(1000)){         // Função que se executa se existir ligação ao fim de 1 segundo
        qDebug() << "Initialized!!";            // Caso haja ligação, avisar o utilizador
        socket->write("login administrator admin\r\n");  // Fazer login na gMA2 pela linha de comandos da mesa

        for(int i=0; i<100; i++){               // Ciclo que escreve "Executor 1.1 at 0"
            QString text = "exec 1.1 at ";      //   até "Executor 1.1 at 100", incrementando um valor
            text.append(QString::number(i));    //   por cada ciclo.
            QString cr = "\r\n";
            text.append(cr);
            QByteArray bt = text.toLatin1();
            const char *txt = bt.data();

            socket->write(txt);
            socket->waitForBytesWritten();
            QThread::currentThread()->msleep(50);
        }
    }
    else                                        // Caso não exista ligação, avisa o utilizador dessa situação
        qDebug() << "Cannot connect";

    socket->disconnectFromHost();               // No final, desligar da grandMA2
}
```

Figura 19 - Excerto do programa de teste de comunicação via Telnet. Escrito em C++

Na grandMA2, após criar uma sessão e habilitar a ligação Telnet nas definições, foi possível ver os comandos recebidos após a execução do programa teste (Anexo 1).

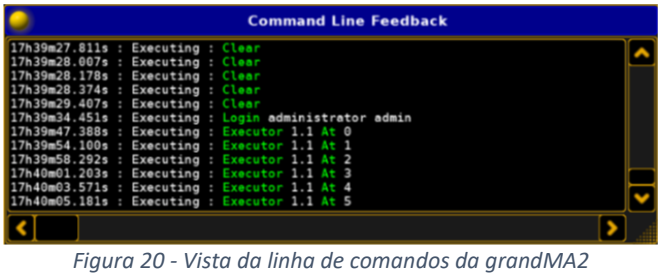


Figura 20 - Vista da linha de comandos da grandMA2

[11] O caracter duplo "//", significa início de um comentário. Na altura da execução o compilador ignora tudo o que está à direita desses caracteres.

- Teste de comunicação via PSN

O primeiro teste PSN foi através da execução do programa demo que vinha associado à biblioteca oficial. Esse demo simulava o sistema solar com 9 planetas a rodar à volta do sol. A posição dos planetas e sol eram enviados para o endereço de *localhost*[12].

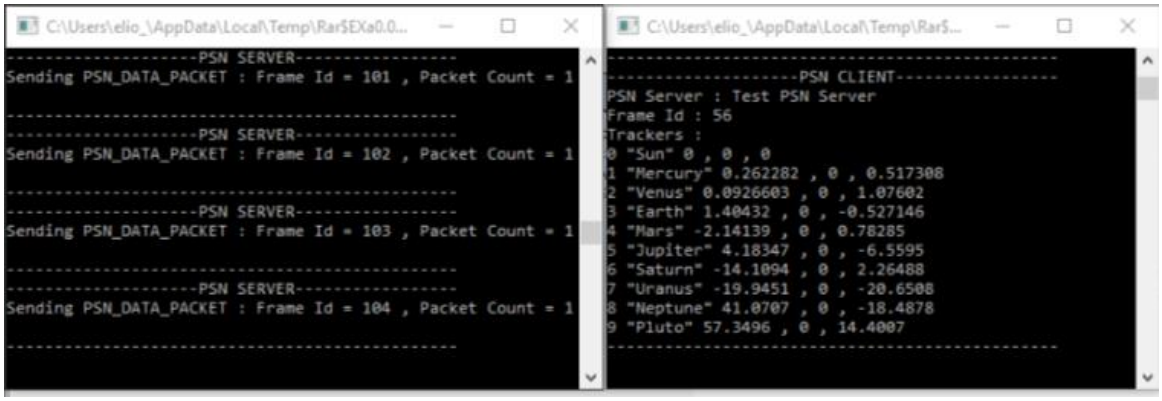


Figura 21 – Vista dos programas de exemplo de servidor e cliente PSN

Foi necessário proceder à alteração do IP no código fonte de forma a coincidir com o da grandMA2 para conseguir visualizar os dados recebidos. Para habilitar essa visualização é necessário navegar até *Setup -> PSN Network Configuration* e clicar em *Enabled* para automaticamente surgir a listagem de fontes de PSN disponíveis:

Figura 22 - Vista de fontes disponíveis no menu "PSN Network Configuration" da grandMA2

Para a visualização dos *trackers* é importante fazer em primeiro lugar o *patch* de um ou mais *stage markers* para assim habilitar a visualização. Sem esse *patch* feito apenas é possível a visualização das fontes de PSN.

Figura 23 - Vista de trackers disponíveis de uma dada fonte PSN

[12] Indica a localização do sistema que está a ser usado. Possibilita a comunicação interna de aplicações.

- Teste de comunicação via TCP

O teste de comunicação via Telnet, que resultou na simples manipulação TCP, foi bastante enriquecedor para o desenvolvimento desta parte pelo que, o programa em Python seria o mais complicado a desenvolver. Assim sendo, depois de alguma pesquisa e tentativas foi possível perceber que afinal seria necessário adicionar 6 linhas a uma parte do código em Python cedido pelo fornecedor. Abaixo exemplifico as linhas extra adicionadas ao código em Python:

```
import socket                                          # Inicialização de sockets para comunicação TCP
from _codecs import utf_8_encode                       # Inicialização de codecs para codificação padrão de mensagens


cliente = socket.socket(socket.AF_INET, socket.SOCK_STREAM)   # Criação de variável “cliente” responsável pela ligação TCP
cliente.connect((“2.0.0.1”, “1234”))                   # Iniciar ligação: IP do Rasperry: 2.0.0.1; Porta utilizada: 1234

strPos = “111,222,333”                                 # Exemplo de variável com dados de posicionamento x,y,z

cliente.sendto(strPos.encode() , ("2.0.0.1",1234) )    # Codificar e enviar mensagem para IP e porta definida
```

Figura 24 - Excerto do programa de teste de comunicação via TCP – parte do Raspberry. Escrito em Python

Assim sendo, a parte escrita em C++ seria um servidor que iria receber os dados do Raspberry que se comportava como cliente. Abaixo está a parte principal desse programa e a sua descrição detalhada:

```
void remoteServer::start()                              // Função que espera por ligação de qualquer
{                                                       //    IP na porta 1234 e processa a mensagem.
    server= new QTcpServer();                           // Inicialização de um novo servidor TCP

    if(!server->listen(QHostAddress::Any, 1234)){       // Caso haja algum erro interno, abortar
        qDebug() << " Error Initializing";
        server->~QTcpServer();
        emit state(0);
        return;
    }
    else{                                               // Caso não haja erros de inicialização,
        qDebug() << "Waiting for clients";              //   avisar o utilizador e
        emit state(1);                                  //   enviar SIGNAL para função principal
    }

    //*************************************
    do{                                                 // Ciclo que cria novo socket cada vez que é
        clientConnected=false;                          //   detetado um novo cliente
        if(server->waitForNewConnection(500)){
            clientConnected=true;
            socket = new QTcpSocket;
            socket = server->nextPendingConnection();
            connect(socket, &QAbstractSocket::disconnected, this, &remoteServer::clientDisconnected);
            qDebug() << "New Pozyx Client";
        }
    }while(!clientConnected && !mStop);

    //*************************************
    while(!mStop){                                      // Ciclo que só pára se o utilizador enviar
                                                        //    sinal "mStop" para esta função
        if(socket->waitForReadyRead(2000)){             // Condição que é executada caso haja alguma
            waiting=false;                              //    mensagem pendente.
            QString socketData = socket->readAll();     // Inicializar variável temporária, portadora
                                                        //    da mensagem

                                                        // Partir a mensagem a cada virgula e
            insideTrackers[0].hID = "0x6066";           //   armazenar na variável "insideTrackers"
            insideTrackers[0].posX = socketData.section(',',0,0).toFloat();
            insideTrackers[0].posY = socketData.section(',',1,1).toFloat();
            insideTrackers[0].posZ = socketData.section(',',2,2).toFloat();

            emit trackersData(insideTrackers);          // Emitir essa variável para qualquer slot
        }
    }

    server->~QTcpServer();                              // Após recebido o sinal "mStop", o programa
    mStop = false;                                      //   destrói as variáveis criadas e repõe
    running=false;                                      //   todos os estados iniciais
    clientConnected = false;
    validSocket=true;
    qDebug() << "Pozyx server stopped";                 //   e informa o utilizador
    return;

}
```

Figura 25 - Excerto do programa de teste de comunicação via TCP. Escrito em C++

- Teste de *multithreading*;

Dado que o programa teria de processar dados de posicionamento, transforma-los em PSN, comunicar com a mesa de luz via Telnet e ainda fazer todos os cálculos necessários para a interação, o melhor método de programação possível seria encapsular esses módulos, ou seja, dividir essas tarefas por *threads*[13]. Assim, através das bibliotecas nativas do Qt desenvolveu-se um programa cuja função principal era colocar uma outra em execução numa *thread* diferente. Utilizou-se o programa de teste PSN como teste *multithreading*:

[13] Divisões virtuais de cada núcleo de uma unidade de processamento.

```
myPSN = new workerPSN();            // Criar novo objeto PSN
psnThread = new QThread;            // Criar nova thread virtual
myPSN->moveToThread(psnThread);     // Mover o objeto para a thread
psnThread->start();                 // Iniciar a thread
```

Figura 26 - Excerto do programa de teste de multithreading. Escrito em C++

- Teste do sistema Pozyx

Assim que o pack Pozyx ficou disponível, procedeu-se à instalação do sistema e à execução dos exemplos cedidos. Com esses exemplos foi possível verificar a posição da *tag* através da linha de comandos. Posteriormente integrou-se o Raspberry Pi com o programa de teste modificado para enviar mensagens TCP.

Com o Raspberry a enviar os dados de posicionamento via TCP para o programa de teste, foi concluído o inicial problema de comunicação entre régie e palco.

O sistema Pozyx vinha com a versão de *firmware* 1.0, em que a velocidade de atualização era cerca de 20 e poucos hertz. Assim que se procedeu à atualização para a versão 1.1 surgiu um grande entrave ao uso das novas funcionalidades. No sistema, existe a possibilidade de obtenção de localização da *tag* a partir da própria *tag*, objetivando o acoplamento um Arduino, ou remotamente a partir de uma âncora à escolha. Antes da atualização, ambas as possibilidades foram testadas e estavam a funcionar corretamente. No entanto, após atualização apenas era possível usar o modo direto, pelo que remotamente os dados de localização recebidos eram nulos ou causadores de erros. Foram feitas diversas tentativas de atualização do *firmware* e inclusive estabeleceu-se contacto com a equipa de desenvolvimento, mas mesmo assim não foi possível a sua resolução. Lamentavelmente, a única solução foi usar a versão 1.0 e otimizar a atualização e precisão, que veio a manter-se nos 20Hz.

3.4.2.1 Filtro de Kalman

Logo nos primeiros testes foi possível verificar que as medições não eram tão suaves quanto se esperava, dando a sensação de que a *tag* "saltava" de posição em posição, mesmo estando parada em cima de uma mesa (Anexo 2). Isso deu início à pesquisa de métodos que suavizassem as medições, surgindo assim o filtro de Kalman.

O filtro de Kalman é um método matemático criado para reduzir as incertezas de uma medição ao longo do tempo, possibilitando ainda uma previsão do próximo valor a ser medido com base nas anteriores. É um método bastante complexo que utiliza cálculos de matrizes algébricas de grandes dimensões e possibilita a fusão de dados como posição e velocidade, para calcular uma previsão mais exata.

Neste caso, foi usada a posição e velocidade tridimensionais para suavizar as medições. Utilizou-se para o efeito uma biblioteca *open-source*, a Eigen, que possibilita a aplicação de funções algébricas com bastante facilidade. Com este filtro introduzido as vantagens verificaram-se instantaneamente. Utilizou-se uma *tag* associada a um motor elétrico *stepper*[14] para movimentar a *tag* uniformemente. No seguinte gráfico pode verificar-se a diferença da medição da posição em X, em milímetros, para a filtrada:

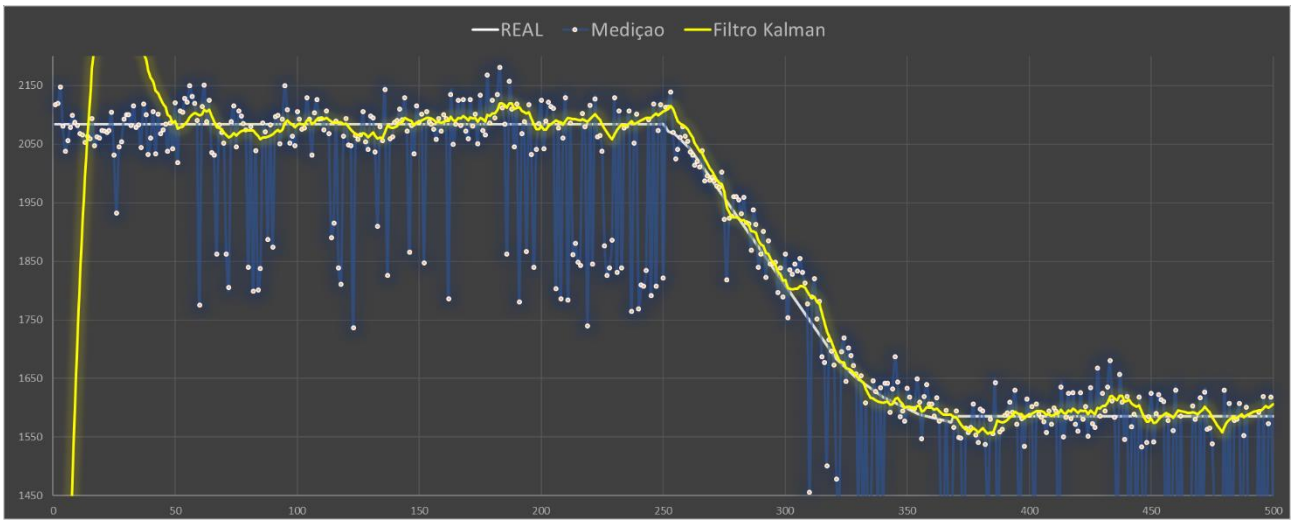


Gráfico 2 - Análise de medições com o filtro de Kalman

3.4.3 Desenvolvimento de hardware adicional

Até ao momento todos os testes tinham sido feitos com a *tag* ligada ao transformador que veio associado, estando sempre dependente de alimentação por cabo. Para tornar a *tag* portátil foi adicionada uma bateria LiPo de 1500mAh que fornecia 3.7V a um *step-up voltage regulator*[15] que por sua vez alimentava a *tag* com os requeridos 5V. Após testes básicos de funcionamento com a bateria foi idealizada uma caixa personalizada, recorrendo à impressão 3D, resultando o produto que mostra a imagem à direita.

Figura 27 - Primeiro protótipo de caixa 3D

Os anexos 10 e 11 mostram partes dessa criação.

[14] Motor elétrico com possibilidade de movimentação passo-a-passo.
[15] Regulador de voltagem que amplifica a voltagem de entrada

Figura 28 - Segundo protótipo de caixa 3D

Após alguns testes verificou-se que era possivel ver os leds em funcionamento mesmo com a caixa fechada devido à cor e transparência do material utilizado e, que a caixa não fora propriamente pensada para se transportar junto ao corpo, por vezes incomodando o utilizador. Foi então desenvolvida uma nova caixa relativamente mais pequena e aperfeiçoada como mostra a imagem à esquerda.

Enquanto se fazia desenvolvimentos no sistema, foi possível verificar que a bateria conseguia aguentar mais de 12h em funcionamento contínuo (rastreamento intercalado com modo standby).

3.4.4 Desenvolvimento avançado e GUI

Dado que os módulos estavam praticamente testados, seria uma questão de tempo até que tudo se integrasse no mesmo software. Para aumentar a robustez do programa foram ainda inseridas funcionalidades de verificação de dados, verificação de estado das ligações e funções de feedback para o utilizador. Como ainda faltava bastante tempo para a apresentação do projeto, desenvolveu-se parte da GUI que nesse momento era apenas constituída por botões que ligavam ou desligavam os módulos atrás explicados.

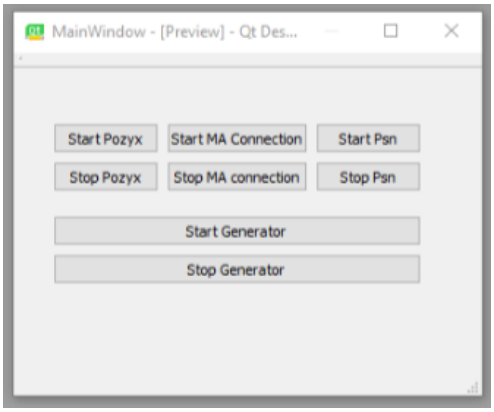


Figura 29 - GUI inicial

Ao desenvolver uma forma de visualização das áreas incorporou-se bibliotecas gráficas que vieram remodelar toda a parte de interativa do programa, que era algo que ainda não estava bem definido a nível de programação.

Concluiu-se que, em vez do processamento de interações se processar num módulo à parte da interface gráfica, iria afinal ser processado no mesmo módulo, integrando a interface gráfica com a interatividade, devido a algumas incompatibilidades de programação que não permitiam que se utilizasse algumas funções fora deste módulo. De forma a dar controlo ao utilizador de todos os pequenos detalhes do sistema o programa ganhou cerca de 10 vezes mais tamanho relativamente à ausência de GUI. Foram ainda adicionados dois módulos de leitura e escrita de dados que permitem guardar cada show criado e abri-lo mais tarde sem perdas de dados.

O software foi reprogramado de forma modular e *multithreading*, o que significa que a operação, velocidade de operação ou recursos utilizados por qualquer um dos módulos não interferem com o resto dos módulos.

A partir daí desenvolveu-se a parte de interação com bastante mais complexidade e rigor. Do anexo 3 ao 6 podemos verificar alguns testes experimentais. Adicionou-se ainda a possibilidade do sistema reagir à orientação do individuo (Anexo 7 com exemplo) ou mesmo à velocidade com que este roda na horizontal, acessando os dados do IMU incorporado, com a desvantagem de reduzir ainda mais a frequência de atualização do sistema, baixando para cerca de 14Hz. Para compreender melhor as possibilidades de interação foram feitos esboços para os 3 inputs possíveis:

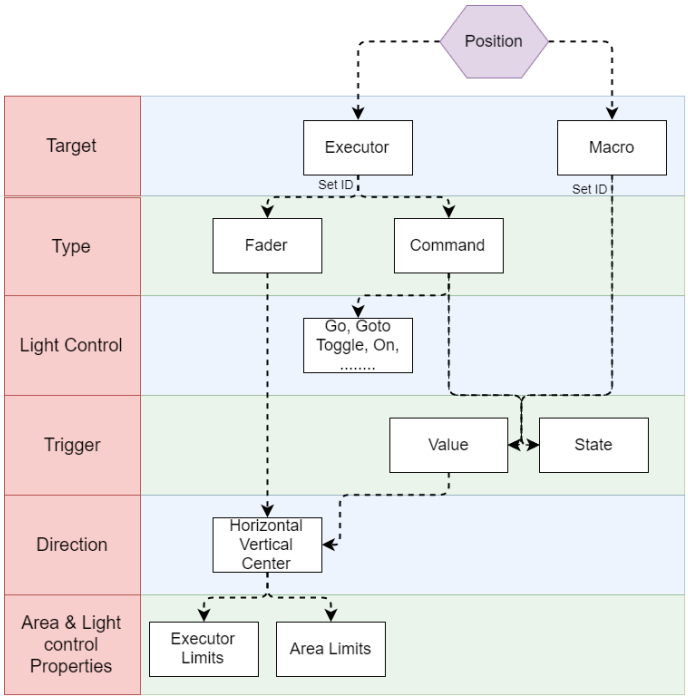


Diagrama 4 - Atributos disponíveis através do input de posição

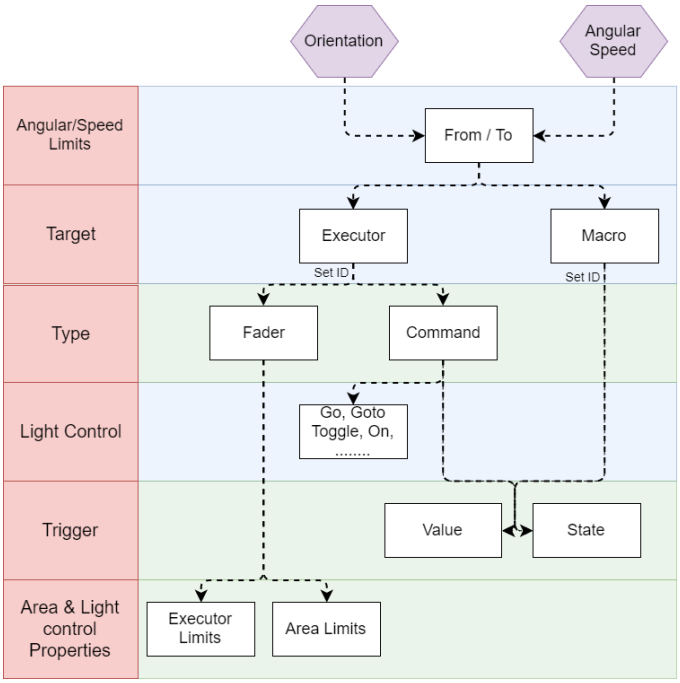


Diagrama 5 - Atributos disponíveis através do input de orientação e velocidade angular

Assim, é possível movimentar *faders*, carregar virtualmente em qualquer botão de qualquer *Executor* e lançar *Macros* conforme a posição, orientação ou velocidade do utilizador.

Pode ainda ser escolhido o *trigger* que desencadeia a execução de *Macros* ou botões de executores. A execução desses controlos pode ser feita por *state*, que é consoante o estado de uma área, ou seja, mal o individuo entre na área, essa área fica ativa e é executado o comando. Caso o input seja a sua posição e o *trigger* definido por *value,* é necessário inserir a que distancia o comando se vai executar. Caso o input seja a sua orientação ou velocidade angular, a definição do *trigger* como *value* define a que ângulo ou velocidade se executa esse comando.

3.4.5 Aplicação das funcionalidades de interatividade

Estas funcionalidades possibilitam que a reação aos estímulos de posição, orientação ou velocidade de rotação possam ser totalmente programados na MA2, enquanto que o software vai servir de interface moduladora entre esses estímulos e a reação programada.

As aplicações são inúmeras, sendo mesmo impossível de enumerar especificamente o que pode ser feito, visto controlar a mesa de luz. No entanto, enumero algumas possibilidades:

- Mudar de cor de um ambiente para outra, à medida que o performer se aproxima de um objeto estático. É possível através da criação de uma área virtual na forma de circulo, com a direção centro definida, que traduz a distância da *tag* ao centro dessa área nos valores mínimo e máximo de um *fader*.

- Dado um cenário com múltiplas divisões, ligar e desligar luz como se tratasse de interruptores. É possível através da associação individual das áreas dessas divisões a *macros* pré-programadas na mesa de luz. É lançada uma *macro* conforme a entrada ou saída das áreas;

- Conforme a performer se vira, iluminar apenas a sua parte frontal. É possível através dos dados de orientação associados a uma área. Esses dados podem ser divididos conforme os ângulos e assim associar cada parte a um *fader*;

- Aumentar a iluminação geral do palco conforme a velocidade com que a performer roda sobre si. É possível através da associação da velocidade angular a um *fader.*

Além destes existem muitos outros exemplos que podem ser efetuados com a incorporação deste sistema. Podemos também considerar uma vantagem o funcionamento colaborativo com a mesa de luz, pois esta já está preparada para o controlo de interfaces externas através de protocolos como o Art-net, MIDI e outros. Assim, é possível o controlo indireto de interfaces de som, vídeo ou outros, dando a possibilidade extra de por exemplo controlar o volume, lançar uma música, mudar a opacidade ou controlar efeitos de vídeo, ou mesmo a criação de espacialização sonora virtual como foi testado através do software Touchdesigner (Anexo 12), conforme os inputs de posição, orientação e velocidade de rotação.

3.4.6 Configurações adicionais

Numa tentativa de aumentar o rigor de comunicação do sistema, foi trocado o método de transporte de dados de localização entre o Raspberry Pi e o programa. De TCP passou a ser utilizado o User Datagram Protocol (UDP), que é um protocolo tão usado quanto o TCP mas cujas características são por vezes menos desejáveis. O UDP possui um grau de confiabilidade mais baixo, dado que não existem garantias de que todos os pacotes vão chegar ao seu destino. Neste caso em concreto, foi a possibilidade de *broadcast* e a velocidade superior do UDP que levaram à troca de protocolo, melhorando desta forma a comunicação entre dispositivos. Apesar do grau de confiabilidade ser menor, mesmo que 1 dos 14 pacotes de dados não fosse recebido num determinado segundo (14Hz), os outros 13 garantiam o funcionamento do sistema, pelo que basta para assegurar o bom funcionamento do sistema.

Para que as coordenadas e orientação do programa estivessem em sintonia com a MA, foi posteriormente adicionado um Menu de calibração que permitia alinhar a origem do espaço tridimensional da MA com a origem do espaço bidimensional do programa desenvolvido. Para usar o modo automático, teria de se posicionar a *tag* no centro do palco e carregar no botão "Auto" para definir esse ponto como a origem e automaticamente os valores tridimensionais calculados aparecessem na janela e pudessem ser sujeitos a pequenas alterações pelo utilizador, caso fosse necessário.

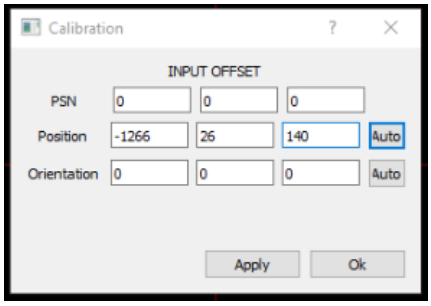


Figura 30 - Menu de calibração

Foi ainda criada uma janela com três abas de configuração, *Tracking*, MIDI e MA2, que possibilitavam mudar atributos do programa como é explicado abaixo. A aba "*Tracking*" foi criada mas até ao momento não possuía qualquer função.

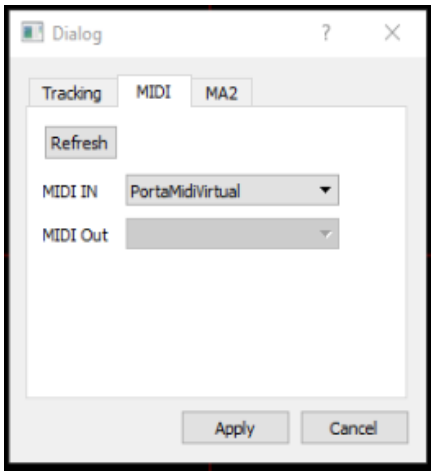


Figura 31 - Menu de Configurações. Secção de MIDI

Na opção *MIDI* seria possível selecionar a porta MIDI de entrada desse mesmo protocolo. Inicialmente foi pensada a opção de inclusão de Art-net de forma a poder controlar o programa através de canais DMX mas essa ideia foi deixada em standby devido à dificuldade de utilização das bibliotecas existentes, utilizando para esse efeito o protocolo MIDI. Com isto, incorporou-se a possibilidade de alternar entre Layers de áreas através da incorporação de um modulo MIDI que recebia notas que se traduziam diretamente no *layer* selecionado. Se por exemplo o programa recebesse a "Nota 1 On", isto traduzia-se na seleção do *layer* 1 e assim sucessivamente.

Para facilitar a ligação à MA2 foi ainda adicionada uma secção que possibilitava a inserção de dados de login de utilizador da grandMA2 e a identificação IP a que a mesa de luz estaria atribuída, para facilitar a configuração de comunicação via *Telnet*.

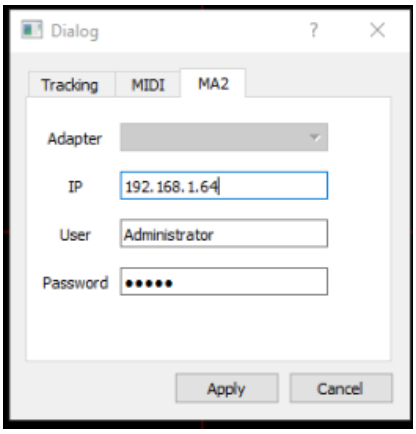


Figura 32 - Menu de Configurações. Secção de MA2

3.4.7 Estrutura Final

No final, o software possuía um estimado leque de funcionalidades:

- Possibilidade de guardar e abrir *shows*;
- Possibilidade de definir as dimensões do palco;
- Possibilidade de adicionar retângulos, círculos e triângulos como áreas de interação;
- Possibilidade de obter a posição, orientação horizontal e velocidade angular horizontal das *tags* e utiliza-las como input de interação;
- Possibilidade de alterar das dimensões, posição, orientação, opacidade ou funções das áreas durante uma performance;
- Envio de dados PSN é praticamente direto;
- Calibração individual da posição para PSN ou software;
- Calibração de orientação;
- Calibração automática de posição e orientação;
- Controlo de *layers* manual ou por MIDI;
- Possibilidade de controlar qualquer *Executor* e *Macro* da MA2 através dos inputs disponíveis;

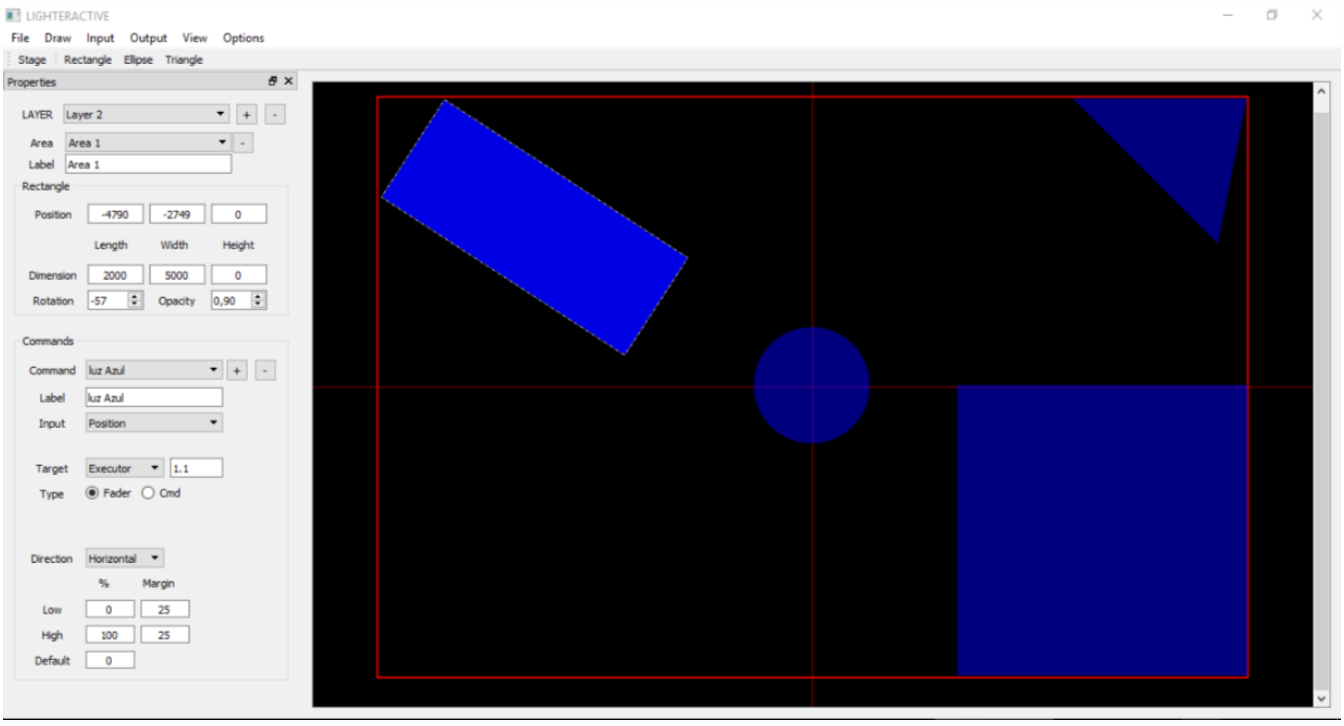


Figura 33 – GUI final do software desenvolvido

Essas funcionalidades são todas possíveis graças a uma estrutura de software final modular, mas sólida, representada pelo seguinte esboço:

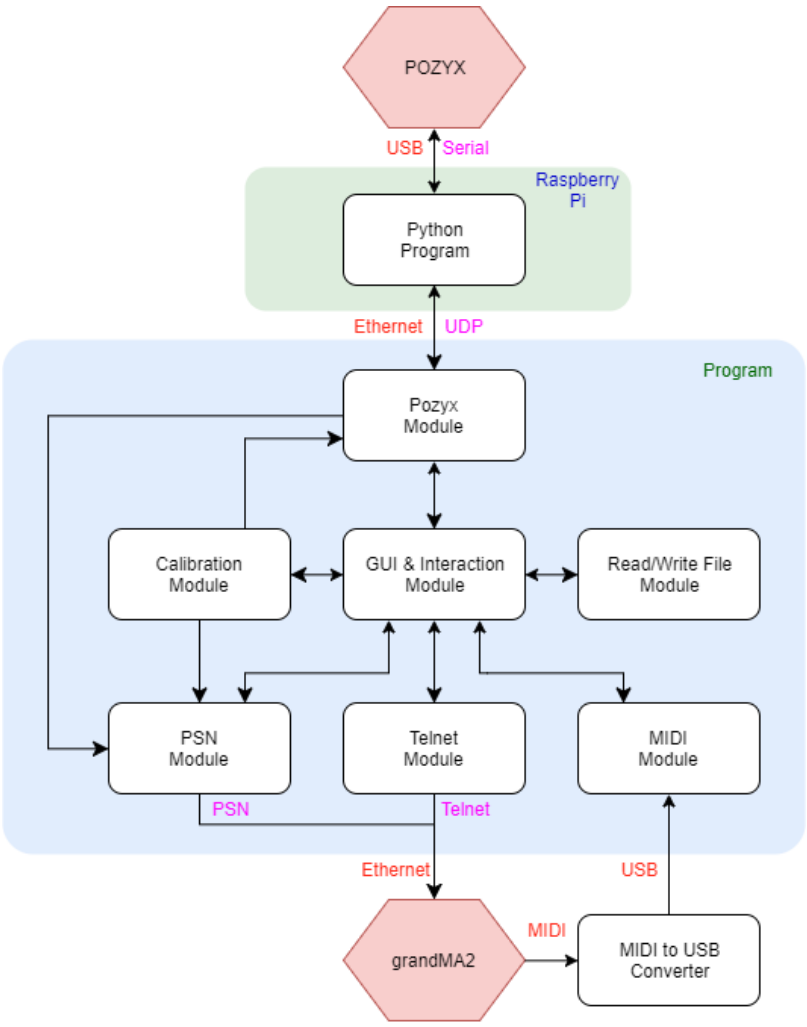


Diagrama 6 - Representação final das principais vias de comunicação e funcionamento do sistema

4. Aplicação

Após o desenvolvimento do sistema de rastreamento e interação foi necessário colocar à prova todo o conceito na forma de performance. Para isso, aliou-se este sistema ao projeto "Light & Body Jam" cujo conceito artístico, desenho de luz e performativo foi concebido pelo designer de luz Pedro Moreira Cabral de forma a alcançar os objetivos do sistema de rastreamento.

Baseando-se num conceito de luz interativa, a luz teria, além da função convencional de iluminação de cena, a função comunicativa. O objetivo performativo seria a criação de um diálogo entre o performer, a luz e o operador de luz durante a performance, como se pode constatar na descrição:

> *E se o corpo marcasse a luz em vez de ser marcado por ela?*
> *Se o corpo submetesse a luz em vez de se submeter a ela?*
> *Foi este o ponto de partida desta performance…*
>
> *"Light & Body Jam" nasce da necessidade de aplicação do sistema de interação criado no âmbito do Projecto Final de Mestrado em Design de Luz. Este projeto de mestrado visa o desenvolvimento de novas tecnologias, técnicas e ferramentas de interação direcionadas para o profissional de iluminação. Tracking e interação são os principais conceitos deste projeto que visa viabilizar todo um novo leque de possibilidades de iluminação, tanto no processo criativo, como na operação ao vivo.*
>
> *Aliando a tecnologia às artes performativas, "Light & Body Jam" vem criar a ponte entre a luz, performer e designer/performer de luz, de forma extremamente peculiar.*
>
> *Esta performance parte da luz enquanto material, gerador de estímulos para a mente, tentando evocar memória emocional trazida para o espaço real através do corpo. Com história no flamenco, este corpo vem carregado de vivências que tentarão ser evocadas da memória através da luz. Por sua vez, os movimentos resultantes desta resposta emocional serão eles próprios estímulos ao performer de luz que reagirá com um novo impulso de luz iniciando-se uma comunicação em retroalimentação que se prevê efémera e cheia de intensidade. A proposta é, então, uma jam session para luz e corpo dentro de uma estrutura com muito espaço para a improvisação.*

4.1 Montagem e calibração

Nesta etapa, a montagem de luz e cenário tiveram prioridade e só após a montagem da maior parte desses dois é que foi decidido onde ficariam as quatro âncoras. Inicialmente foi idealizada a montagem das âncoras nas próprias varas de iluminação, mas como este é um sistema protótipo, determinou-se que o melhor seria a colocação delas em locais facilmente acessíveis para facilitar o despiste de problemas.

Dado que praticamente toda a envolvência do palco foi usada como parte integrante da performance, foi medida a distancia entre extremos da área performativa, concluindo que seria à volta de 13,5 metros de largura por 8,5 metros de profundidade, que perfazem uma área de 114m^2, valor ligeiramente inferior ao limite máximo do sistema POZYX (120m^2).

Com isto, foi efetuado um teste inicial para testar a viabilidade do sistema com uma disposição de âncoras de forma irregular, ao contrário do que aconteceu em praticamente todos os testes feitos anteriormente. Este primeiro teste visou a colocação da ancora que fica ligada ao Raspberry perto da régie, para despistar problemas de ligação ou erros inesperados que necessitassem de reiniciar a ligação USB durante os ensaios ou performance. Pouco depois concluiu-se que esta não seria a melhor forma de montagem do sistema devido à grande distancia dessa ancora com a performer e principalmente com outras âncoras que chegava a ser superior a 13 metros, ultrapassando a área limite do sistema, inviabilizando os dados de rastreamento ou mesmo impossibilitando a ligação entre dispositivos.

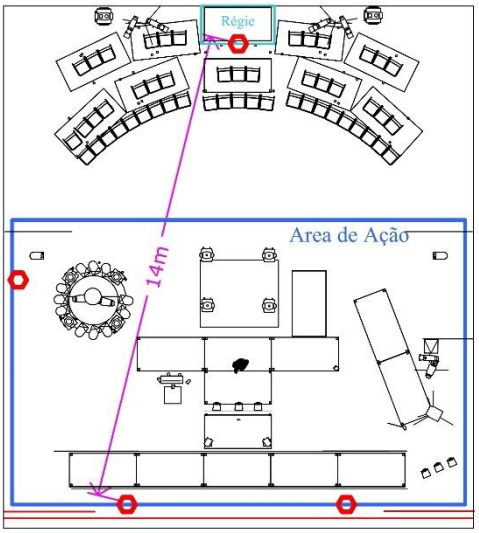


Ilustração 4 - Esboço de localização inicial das âncoras (hexágonos a vermelho)

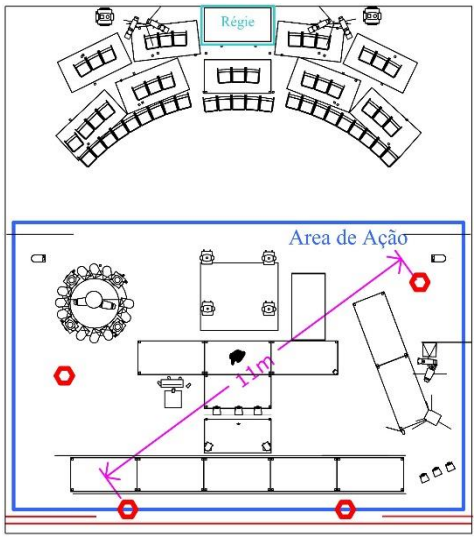


Ilustração 5 - Esboço da nova localização das âncoras (hexágonos a vermelho)

Foram então movimentadas as âncoras para localizações mais viáveis, perfazendo aproximadamente a forma de um trapézio, no qual a distancia entre as duas âncoras mais distantes seria de 11 metros, ficando a área de operação do sistema dentro do limite máximo do sistema Pozyx e alcançando toda a área performativa.

Após novos testes verificou-se ainda assim alguma instabilidade do sistema, verificando-se inconsistências nos dados de localização da *tag* que ocasionalmente "saltavam" para pontos distanciados alguns metros da posição real da *tag*. Estes saltos duravam menos de meio segundo, mas seria suficiente para se notar, principalmente na perseguição de luz. As âncoras foram aproximadas cerca de meio metro a um metro em direção ao centro da área performativa, mas ainda assim o sistema fornecia dados incorretos, apesar de uma menor frequência.

Essas incoerências de localização não eram lineares nem pareciam estar associadas a alguma área do palco em especial. Adicionando a falta de tempo *in loco* para despistar outro tipo de causas prováveis, como a interferência do corpo humano ou o excesso de materiais metálicos, determinou-se a utilização do protótipo com esse risco associado.

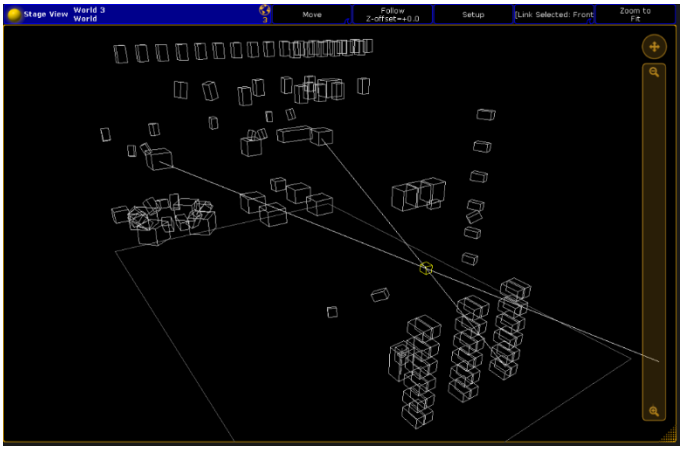


Figura 34 - Visualização do rastreamento de luz na grandMA2. Tag a amarelo

Para que a perseguição de luz pudesse ser ativa, foi primeiro necessário a criação do desenho tridimensional dentro da mesa de luz, para isso convertendo o desenho desenvolvido no WYSIWYG para a MA2. Após esse passo, e para que ficasse pronta a ser utilizada sem problemas, seria necessário a sua calibração. Após o sistema estar montado e a comunicar foi feita a calibração por software das coordenadas para que o ponto de origem do sistema coincidisse com a origem do desenho de luz tridimensional da mesa de luz. Após esse passo concluído percebeu-se que apesar do sistema estar calibrado no software, a perseguição de luz continha um desvio de aproximadamente um metro num dos eixos tridimensionais, dependendo da localização real. A primeira opção a tomar foi a calibração extra dos dados enviados via PSN através

do menu de calibração do software. Ainda assim, a calibração não foi totalmente satisfeita, presumivelmente devido à inserção desses robôs em locais que não coincidiam exatamente com o desenho de luz tridimensional, muitas vezes bastando estar 50cm ao lado para obter diferenças de um metro ou mais na perseguição de luz, dependendo da distancia à *tag*. Nesses casos, foi feita a movimentação desses robôs no desenho tridimensional e os requisitos mínimos de funcionamento da perseguição de luz foram atingidos. O anexo 8 providencia uma parte dessa calibração.

4.2 Programação e operação

A programação de luz da performance foi determinada pelo designer de luz, tendo em conta o percurso que a performer faria. Devido à conclusão do software muito próxima da data da apresentação da performance, não foi possível a instrução total do funcionamento do software para que a performance fosse totalmente programada pelo Pedro. Apesar de ter sido idealizado como ferramenta auxiliar para o programador de luz, a programação da interação foi feita à parte, em função do que era pedido pelo Pedro, resultando numa programação colaborativa da performance.

Para a perseguição de luz, a localização da *tag* era enviada em tempo real para a mesa de luz via PSN, pelo que a ativação desta funcionalidade estava unicamente dependente da programação na mesa de luz. A nível de interação, a programação contou com 8 *layers* de áreas, ou seja, 8 momentos de interações distintas, intercalados com *layers* em branco caso fosse necessária a mutação entre *layers* sem qualquer interação, contando com um total de 16 *layers*. Apesar de existirem 3 tipos de formas, em toda a performance apenas se criou áreas com forma retangular. Foram atribuídos múltiplos comandos a cada área mas apenas foi atribuído o controlo de *faders*, não tendo a possibilidade de ter sido experimentada a atribuição dos comandos de executores como *Go*, *Goto*, *On*, *Toggle*, entre outros, ou mesmo a execução de *Macros*. Apesar disso, tanto o input de posição como o de orientação ou velocidade angular puderam ser atribuídos ao controlo de *faders* com sucesso e sem qualquer problema.

A programação de interações através do software criado é feita de modo colaborativo com a mesa de luz. O software serve como extensão da mesa de luz na medida em que faz uso dos controlos existentes para conseguir uma gestão "inteligente" do estado da iluminação de forma a criar interações. Para isso, foi necessária uma abordagem à programação de luz completamente distinta da forma convencional de que se está habituado.

Por exemplo, para criar a interação entre a performer e os feixes de luz verticais, os quais acompanhavam a performer em todo o seu percurso, ligando o próximo feixe e desligando o anterior à medida que a performer se movimentava, foi necessária a utilização da técnica de *pixel mapping* que a grandMA2 fornece.

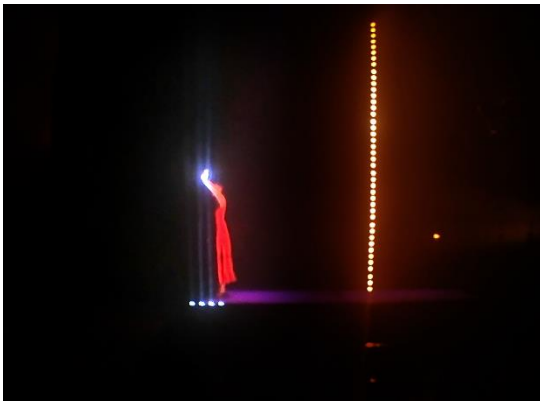


Figura 35 - Interação da posição da performer com os feixes de luz verticais

Através da inserção de uma imagem num *bitmap,* foi possível modular dois pontos fundamentais dessa interação:

- A dimensão horizontal do bitmap é associada à quantidade de feixes de luz ativos.

- A posição horizontal do bitmap é associada à posição da *tag*, através da gravação dos dois limites relativos à posição do bitmap no layout num fader, como vemos na figura abaixo e representação em vídeo no anexo 9.

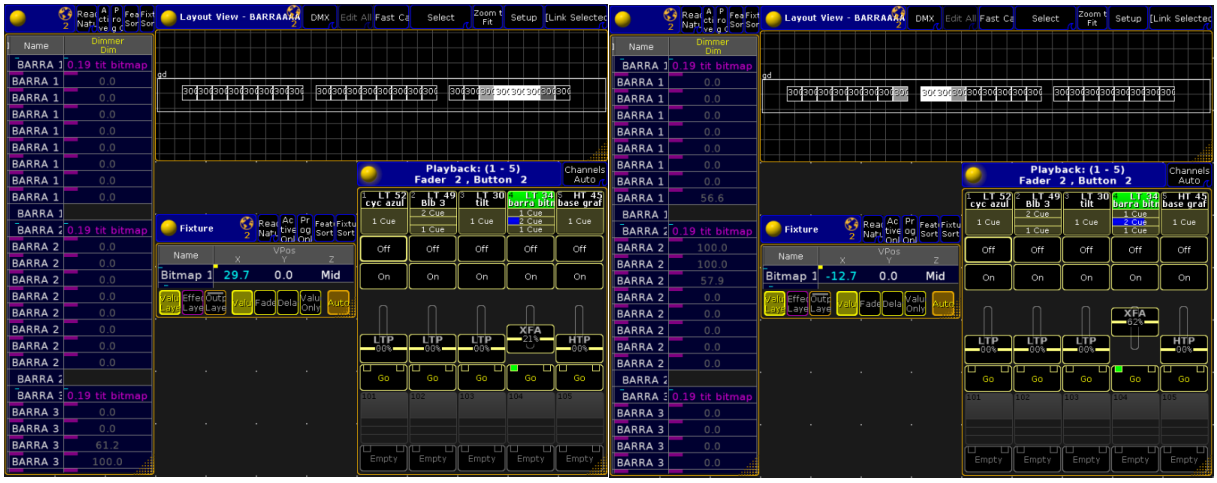


Figura 36 - Programação de interação de barras de led com recurso a bitmap. Fader a 21%, à esquerda e 62% à direita. À esquerda o valor dimmer das barras, ao centro o valor da posição do bitmap, à direita a posição do fader e em cima a visualização de pixel mapping num layout.

5. Análise de resultados

Por ser um projeto protótipo, este sistema abriu a possibilidade de experimentação constante do que funcionaria ou não, tanto na parte de física do sistema, como no software ou na utilização deste pelo utilizador. Ao longo de todo o desenvolvimento surgiram novos requisitos e lacunas, umas óbvias e outras que só na aplicação real do sistema se conseguiu detetar.

A incapacidade de utilização do *firmware* mais recente do Pozyx foi o principal inconveniente do projeto, que acabou por limitar em cerca de ¼ da capacidade real do sistema. Esta limitação agravou-se quando surgiu o requisito de utilização das funcionalidades extra do IMU, fixando a frequência de atualização nos 14Hz. Com esta frequência, surgiram consequências em todas as partes do sistema, como a latência do sistema caso a performer se movimentasse rápido ou o acréscimo de erros de medição. Também assim, ao adaptar-se automaticamente à nova taxa de atualização, o filtro de Kalman deteriorou-se, agravando os "saltos" que se verificavam no início do projeto.

A calibração, que inicialmente se menosprezava, tornou-se bastante complexa na tentativa de efetuar medições no meio de toda a cenografia, equipamento de iluminação, estruturas de apoio e panejamento. Foram necessárias várias tentativas para que as medições fossem adquiridas com o menor erro possível.

Surgiu também um erro de software relativo à mudança de *layers* do programa que por vezes não funcionava à primeira tentativa, mas que seria contornado ao carregar uma segunda vez.

Os maiores inconvenientes surgiram na implementação das interações. Já na parte final do projeto, pouco antes da performance, foi possível verificar que as funções que o software disponibiliza se tornaram pouco práticas quando confrontadas com uma interação mais complexa. Foi necessário a utilização de técnicas de *pixel mapping* com objetivos para o qual essa técnica não foi pensada. Das diversas tentativas de integração do software com a MA2, a técnica de *pixel mapping* pareceu ser a mais indicada para muitas situações, mas que era trabalhosa e necessitava de bastante tempo para detalhar uma só interação como é o caso das barras de leds utilizadas na performance, atrás referidas. Seria necessário detalhar a imagem a utilizar no bitmap, as dimensões e a posição inicial e final do bitmap o que envolvia bastante experimentação para se chegar ao resultado pretendido.

Apesar desses inconvenientes o sistema foi capaz de fornecer os meios para os três objetivos do projeto: rastreamento e interação com a luz de baixo custo. Foi ainda possível a utilização extra da orientação e velocidade angular como ponto de partida de interações pelo que, o balanço de todo o projeto foi bastante positivo.

6. Trabalho futuro

O objetivo deste projeto foi a criação de uma interface-protótipo, catalisador para o desenvolvimento futuro de um produto final, que forneça as ferramentas necessárias para que o profissional de iluminação consiga expandir as formas de criação e desenvolvimento de desenhos de luz de uma forma transversal a todos os elementos de uma performance.

Sendo este um projeto cujo objetivo é a criação de um produto de baixo custo capaz de criar uma comunicação entre o performer, a luz e o profissional de iluminação, a futura intenção é de aprofundar o desenvolvimento do sistema de forma a criar um produto final cujas funcionalidades sejam as mais adequadas e simples possíveis.

O primeiro passo será investigar a forma mais apropriada para criar interações de forma simples e eficaz. Talvez a solução esteja num modelo aproximado do Blacktrax em que a comunicação é feita ao contrário, da mesa para o sistema, ou talvez esteja na criação de uma base de dados de autómatos de iluminação e controlo direto destes. Para a obtenção de uma taxa de atualização superior talvez seja possível a comunicação individual de cada âncora a uma unidade de controlo, em vez da comunicação interna atual. De qualquer forma, a planificação de necessidades e experimentação estão no percurso de qualquer solução que possa surgir durante esse desenvolvimento futuro.

Após esse passo, será necessário otimizar ou mesmo criar novas funcionalidades do sistema:

- Criar uma equipa de desenvolvimento
- Otimizar a frequência de atualização
- Otimizar a comunicação entre dispositivos
- Reduzir tamanho das *tags*, criando um dispositivo de raíz
- Adicionar módulos de comunicação como Art-net e sACN
- Adicionar módulos de comunicação com protocolos de terceiros, como os do Blacktrax
- Desenvolver suportes para âncoras
- Otimizar a organização do código-fonte do software
- Criar ambientes com condições diversificadas para o teste contínuo do sistema
- Recorrer à opinião e experiência de profissionais de áreas performativas de forma a enriquecer o desenvolvimento do sistema

É assim necessária uma investigação profunda de conceitos e opiniões que possam permitir o desenvolvimento do sistema na direção certa para um produto final de sucesso e de baixo custo.

7. Conclusão

Desde a planificação à apresentação, o desenvolvimento integral deste protótipo possibilitou o surgimento de situações peculiares e problemas cuja resolução experimental permitiu o levantamento de questões que dificilmente seriam detetadas apenas com investigação e especulações teóricas. A criação desta ferramenta possibilitou ainda a tradução de ideias e conceitos de interação que eram previamente intraduzíveis. Complementando estas circunstâncias está ainda o facto de que é possível obter este tipo de relação interativa com a luz com relativamente pouco investimento e boa qualidade.

Através do desenvolvimento e apresentação do projeto verificou-se que existem inúmeras possibilidades de interatividade entre um performer e a iluminação. A criação artística, definida na performance apresentada, seria baseada num conceito de luz com características comunicativas, evocando um relacionamento interativo com a bailarina. Dessa forma, na tentativa de fornecer uma ferramenta que possibilitasse esses e outros tipos de interatividade foram idealizadas, desenvolvidas e postas à prova funcionalidades que vieram a tornar-se fundamentais principalmente na altura da programação de uma performance.

Devido ao reduzido leque de ferramentas deste género, o profissional de iluminação não sabe *à priori* com o que pode contar, dando às primeiras performances um conceito mais experimental na criação de conteúdo interativo. Com isso, penso que com a chegada de novas soluções de interatividade deste género podem surgir criações artísticas com grande qualidade que podem traçar novos trilhos na forma em que o performer de luz idealiza e desenvolve o seu trabalho. Ou mesmo criar novos caminhos que levam à comunicação ainda mais profunda com encenadores, performers e todos os outros intervenientes de uma criação artística performativa.

O protótipo apresentado necessitou de cerca de seis meses de desenvolvimento e cerca de 700€ em equipamento para apresentar um mínimo de condições que permitissem a funcionalidade principal do Blacktrax ou do Zactrack, a perseguição de luz, e funcionalidades adicionais de interatividade. Essas funções extra não são possíveis de verificar em praticamente nenhum sistema de rastreamento estudado, salvo o Zactrack, que possibilita alguns efeitos predefinidos, mas que não considero uma interação real, não existe propriamente uma ação-reação. São apenas efeitos de movimento que mudam de posição conforme a referência numa *tag*.

A maior diferença verificou-se especialmente na distinção abismal de custos, em que este protótipo custou cerca de 40 vezes menos que o Zactrack. É claro que os custos de desenvolvimento deveriam entrar nas contas, mas como praticamente todo o projeto foi efetuado por uma pessoa, esse valor torna-se insignificante se pensarmos na venda deste produto no vasto mercado de pequenas e médias infraestruturas na área performativa.

Pode mesmo pensar-se na inclusão de futuros desenvolvimentos e mesmo assim é muito provável que o custo fique mais baixo cerca de 15 vezes o valor do Zactrack ou 26 vezes o do Blacktrax.

Com isso, todos os objetivos do projeto foram atingidos, através de variadas funcionalidades, umas com maior qualidade que outras, e todas a necessitarem de um desenvolvimento mais cuidado de forma a criar um sistema completo, de simples utilização e otimizado para as artes performativas.

8. Referências

Citadas

Akten, M. (2 de 1 de 2017). *Memo Akten - Pattern Recognition (2016)*. Obtido de http://www.memo.tv/pattern-recognition

Backstage, A. (7 de 12 de 2016). *Backstage Academy on VIMEO*. Obtido de https://vimeo.com/user58130656

Frey, T. (25 de Abril de 2013). The futurist: We're watching you. Colorado, Estados Unidos da América.

Hunt, N. (2013). Performing Interventions: a methodology for reinventing the role of the theatre lighting artist through practice-research. *Sixth International Conference of Doctoral Studies of*

Theatre Schools, (pp. 1-11). República Checa. Obtido de http://difa.jamu.cz/english/6th-international-conference-of-doctoral-studies-of-theatre-schools-papers.html

Kent, R. (2011). *A Spin Around Moving Lights.* Tennesse, EUA: High End Systems.

Ramsey, F., & Robet, H. (8 de 12 de 2014). An Analysis of the Accuracy of Bluetooth Low Energy for Indoor Positioning Applications. Em F. Ramsey, & H. Robert, *Proceedings of the 27th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2014)* (pp. 201-210). Tampa, Florida, Reino Unido.

Consultadas

ART. (2017). *ART Advanced Realtime Tracking*. Obtido de http://www.ar-tracking.com/

Bisatto, A. P., & Peres, A. (29 de Outubro de 2015). Localização de Estação sem fio utilizando trilateração. Canoas, RS, Brasil.

CAST Software Ltd. (10 de Agosto de 2017). *BlackTrax Wiki*. Obtido de http://wiki.blacktrax.ca

Coolux - a Christie Company. (21 de Março de 2017). Pandoras box brochure 2017. Cologne, Alemanha. Obtido de http://www.coolux.de/

Edmonds, E. (1 de 08 de 2011). Art, Interaction and Engagement. *15th International Conference on Information Visualisation* (pp. 451-456). Londres: Institute of Electrical and Electronics Engineers (IEEE). doi:10.1109/IV.2011.73

Edmonds, E. (2011). Interactive Art. Em E. Edmonds, & L. Candy, *Interacting: Art, Research and the Creative Practitioner* (pp. 18-32). Oxfordshire: Libri Publishing.

Floreani, L. (24 de Fevereiro de 2015). *Indoor positioning with beacons and mobile devices.* Obtido de http://bits.citrusbyte.com/indoor-positioning-with-beacons/

Hara, S., & Anzai, D. (2008). Experimental Performance Comparison of RSSI and TDOA-Based Location Estimation Methods. *Proceedings of the 67th IEEE Vehicular Technology Conference*, (pp. 2651-2655). Singapura. doi:10.1109/VETECS.2008.581

Institute for Robotics and Intelligent Systems. (14 de Setembro de 2011). *USC Computer Vision -- Major Research Topics and Projects*. Obtido de USC IRIS -- Institute for Robotics and Intelligent Systems: 2011

Kleeman, L. (Janeiro de 1996). Understanding and Applying Kalman Filtering. *Proceedings of the Second Workshop on Perceptive Systems*.

Microsoft. (2017). *Kinect Hardware*. Obtido de Microsoft Developer: https://developer.microsoft.com

OptiTrack. (3 de Novembro de 2016). *OptiTrack - Motion Capture Systems*. Obtido de http://optitrack.com/

Palmer, S. (2013). *Light: Reading in Theatre Practice.* Basingstoke, Reino Unido: Palgrave.

Paul, A. S. (Setembro de 2010). Sigma-point Kalman smoothing: algorithms and analysis with applications to indoor tracking. Oregon, Estados Unidos da América.

Sanda, B. J. (Junho de 2016). Improved Accuracy in Real-Time RFID Localization Systems Using Kalman Filtering. Michigan, Estados Unidos da América.

Universität Siegen. (30 de Janeiro de 2016). *Navigation | Zess - Zentrum fur Sensorsysteme.* Obtido de Universität Siegen: Zukunft menschlich gestalten | Universität Siegen: https://www.uni-siegen.de/zess/profil/navigation.html

Velde, S. V. (3 de 2016). *POZYX - centimeter posisioning for arduino*. Obtido de https://www.pozyx.io/

Vicon. (2017). *Motion Capture Systems*. Obtido de https://www.vicon.com/

Zkoor Technologies. (31 de Março de 2017). *Zkoor Technologies*. Obtido de http://www.zkoor.com/

9. Anexos

Formato digital:

Anexo 1 – Teste Telnet
Anexo 2 - Exemplo de medições com 'saltos'
Anexo 3 - Teste de interação de posição com fader
Anexo 4 - Teste de perseguição de luz com filtro de Kalman
Anexo 5 - Teste de interação entre posição e a cor dos projetores
Anexo 6 - Teste de *tracking* e interação da posição com a cor
Anexo 7 - Teste de interação com input de orientação
Anexo 8 - Calibração manual para *tracking*
Anexo 9 - Uso de bitmap
Anexo 10 - Medições para criação da caixa
Anexo 11 - Criação virtual da caixa para impressão 3D
Anexo 12 - Controlo externo de som através de Touchdesigner

ESCOLA
SUPERIOR
DE MÚSICA
E ARTES
DO ESPETÁCULO
POLITÉCNICO
DO PORTO

P.PORTO

M

MESTRADO
TEATRO
Design de Luz

Sistema de rastreamento interativo para iluminação cénica como ferramenta de criação

Élio Silva Moreira